Parameter Estimation in Wiener-Hammerstein Models

by

Khaled Husain Al-Ajmi

A Thesis Presented to the

FACULTY OF THE COLLEGE OF GRADUATE STUDIES

KING FAHD UNIVERSITY OF PETROLEUM & MINERALS

DHAHRAN, SAUDI ARABIA

In Partial Fulfillment of the
Requirements for the Degree of

MASTER OF SCIENCE

In

SYSTEMS ENGINEERING

April, 1999

INFORMATION TO USERS

This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps. Each original is also photographed in one exposure and is included in reduced form at the back of the book.

Photographs included in the original manuscript have been reproduced xerographically in this copy. Higher quality 6" x 9" black and white photographic prints are available for any photographs or illustrations appearing in this copy for an additional charge. Contact UMI directly to order.

UMI®

Bell & Howell Information and Learning
300 North Zeeb Road, Ann Arbor, MI 48106-1346 USA
800-521-0600

PARAMETER ESTIMATION IN WIENER-HAMMERSTEIN MODELS

BY

KHALED HUSAIN AL-AJMI

A Thesis Presented to the

FACULTY OF THE COLLEGE OF GRADUATE STUDIES

KING FAHD UNIVERSITY OF PETROLEUM & MINERALS

DHAHRAN, SAUDI ARABIA

In Partial Fulfillment of the
Requirements for the Degree of

MASTER OF SCIENCE

In

SYSTEMS ENGINEERING

April 1999

KING FAHD UNIVERSITY OF PETROLEUM & MINERALS

DHAHRAN, SAUDI ARABIA

DEANSHIP OF GRADUATE STUDIES

This thesis, written by **Khaled Husain Al-Ajmi** under the direction of his Thesis Advisor and approved by his Thesis Committee, has been presented to and accepted by the Dean of Graduate Studies, in partial fulfillment of the requirements for the degree of **Master of Science in Systems Engineering.**

THESIS COMMITTEE

H. Shabaik 5/2/99.
Dr. H. E. Emara-Shabaik, **Chairman**

Md. Shahgir Ahmed 5/2/99
Dr. M. Shahgir Ahmed, **Co-Chairman**

5/2/99
Dr. Yousef Abdelmagid, **Member**

5/2/99
Dr. Shokri Z. Selim, **Member**

5/2/99
Dr. Abdulbasit A. Andijani
Chairman, Department of Systems Engineering

Dr. Abdallah M. Al-Shehri
Dean of Graduate Studies
Date : 4-5-99

Dedicated as a humble tribute to

my beloved mother and father

whose prayers, sacrifices, inspiration and love

led to this accomplishment

and to

my loving sisters and brothers

Acknowledgments

First and foremost, all praise is made to Allah, the Almighty, Who gave me every reason to carry out this work. I thank Him for the unlimited favors that He bestowed on me, and seek His mercy and forgiveness.

Acknowledgement is due to *King Fahd University of Petroleum & Minerals* for its support of this research.

I wish to express my sincere gratitude and thanks to my thesis advisor Dr. Hosam Shabaik. His constant encouragement and constructive criticism throughout this research work is highly appreciated. I would also like to extend my appreciation to the co-advisor Dr. M Shahgir Ahmed. His comments, suggestions, guidance, and critique made the work possible. I would like to extend my sincere and special thanks to my committee members Dr. Yousef Abdelmagid and Dr. Shokri Z. Selim for their constant help, guidance, support and valuable suggestions throughout this work. I should also express my thanks to Dr. H. Al-Duwaish for his valuable help and suggestions.

I would like to thank the chairman, the faculty and the staff of the *Systems Engineering* Department for their great support.

Contents

List of Tables

List of Figures

Abstract

Name: **Khaled Husain Al-Ajmi**

Title: **Parameter Estimation in Wiener-Hammerstein Models**

Major Field: **Systems Engineering**

Date of Degree: **April 1999**

The model description for linear systems is relatively straight forward. Modeling nonlinear processes is more involved due to the lack of a common structure for such systems. It has been shown in the literature that the Wiener-Hammerstein models (also known as block cascaded models) can adequately represent a wide range of nonlinear systems. It is the purpose of this work to investigate extending the classical offline and online parameter estimation algorithms as well as methods based on Evolutionary Programming Techniques to estimate the parameters of the Wiener-Hammerstein models. In addition, the properties as well as the convergence of such algorithms are to be investigated.

MASTER OF SCIENCE DEGREE

KING FAHD UNIVERSITY OF PETROLEUM AND MINERALS
Dhahran, Saudi Arabia
April, 1999

خلاصة الرسالة

الاسم : خالد حسين العجمي

العنوان : تقدير المعاملات في نماذج وينر هامرشتين

التخصص : هندسة النظم

تاريخ الشهادة : أبريل ١٩٩٩م.

من المعروف أن الأنظمة الخطية تتميز بسهولة تمثيلها ونمذجتها، الأمر الذي لا ينطبق على الأنظمة غير الخطيـــة، والتي تفتقر إلى التركيب الموحد لتمثيلها. من خلال استقصاء الدراسات السابقة وجد أن نماذج وينر هامرشتين يمكن أن تمثل وبكفاءة العديد من النظم الغير خطية المختلفة. ولذلك فإن الغرض من إجراء هذا البحث هو اســـتقصاء تطبيـــق خوارزميات تقدير المعاملات، والمستخدمة في النظم الخطية لاستنباط معاملات نماذج وينر هامرشتين. تمـــت في هـــذا البحث أيضا دراسة خصائص ومميزات هذه الخوارزميات.

بالإضافة إلى ما سبق فقد تمت دراسة طرق تقدير المعاملات والمعتمدة على الخوارزميات الجينية في حالة اســـتخدام نماذج وينر هامرشتين لنمذجة النظم الغير خطية.

ماجستير في العلوم

جامعة الملك فهد للبترول والمعادن

الظهران – المملكة العربية السعودية

أبريل ١٩٩٩ م

Nomenclature

Notations

t	time variable
$u(t)$	input signal
$y(t)$	output signal
$\hat{y}(t)$	predicted output signal
$e(t)$	white noise (a sequence of independent random variables)
λ^2	variance of white noise
q^{-1}	backward shift operator
θ	parameter vector
$\hat{\theta}$	estimate of parameter vector
θ_0	true value of parameter vector
n_θ	dimension of parameter vector
N	number of data points
V, J	cost function
$\gamma(t)$	gain sequence
λ	forgetting factor
Λ	covariance matrix of innovations
$\psi(t)$	negative gradient vector
$e_{LS}(t,\theta)$	least squares error corresponding to the parameter vector θ
$e_{OE}(t,\theta)$	output error corresponding to the parameter vector θ
$\varepsilon(t,\theta)$	prediction error corresponding to the parameter vector θ
$Z(t)$	vector of instrumental variables
k	generation index in genetic algorithms

$P(k)$	population in the kth generation
p	population size
f	fitness function
f_n	normalized fitness function
F	cumulative normalized fitness function
P_p	perturbation probability
P_c	crossover probability
P_m	mutation probability
s	reproduction rate
I	identity matrix
A^T	transpose of the matrix A

Abbreviations

ARMAX	AutoRegressive Moving Average with an eXogenous input
NARMAX	Nonlinear AutoRegressive Moving Average with eXogenous input
ANN	Artificial Neural Network
G-model	General model
IGBT	Insulated Gate Bipolar Transistor
OE	Output Error method
PE	Prediction Error method
NLS	Nonlinear Least Squares method
IV	Instrumental Variable method
ROE	Recursive Output Error method
RPE	Recursive Prediction Error method
RNLS	Recursive Nonlinear Least Squares method
RIV	Recursive Instrumental Variable method

SA	Stochastic Approximation method
GAs	Genetic Algorithms
GWN	Gaussian White Noise
UWN	Uniform White Noise
PR	Pseudo-Random signals
PRBS	Pseudo-Random Binary Signals
PRMS	Pseudo-Random Multilevel Signals
MPRMS	Modified Pseudo-Random Multilevel Signals
FSR	Feedback Shift-Register
SSE	Sum of Squared Error
NSR	Noise-to-Signal Ratio
MNE	Mean Normalized Error
ASD	Average Standard Deviation
AIFC	Average Iteration For Convergence

Chapter 1

MODELING OF NONLINEAR DYNAMIC SYSTEMS

1.1 Introduction

The field of system identification can be defined as modeling dynamic systems from experimental data. A dynamic system can be described as in Figure 1.1. The system is driven by input signals $u(t)$ and disturbances $e(t)$. The user can have a control over $u(t)$ but not over $e(t)$. The output signals $y(t)$ are variables that provide useful information about the system. For a causal dynamic system the control action (the input) at time t will influence the output at all subsequent instants of time, i.e. $s \geq t$.

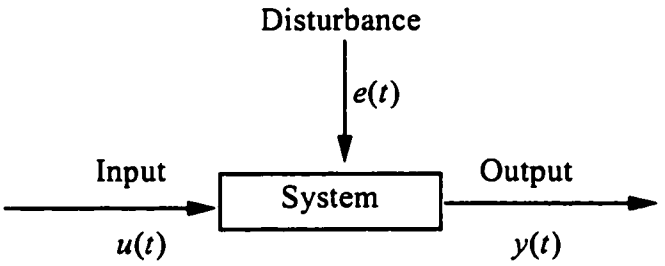


Figure 1.1. General configuration of a dynamic system.

The production of paper, iron, glass or chemical compounds, as well as many other industrial processes, must be controlled in order to run safely and efficiently. A process model is needed in order to design controllers. The models can be of various types and degrees of sophistication.[1]

It is not only the purpose of modeling to aid in design and control. The knowledge of a model can itself be the purpose. If the models can explain measured data satisfactorily, then they may also be used to explain and understand the observed phenomena. In a more general sense, modeling is used in many branches of science as an aid to describe and understand physical phenomena.

Models of dynamic systems can be of many kinds, including the following: [1]

- *Mental, intuitive or verbal models*. For example, this is the form of 'models' we use when driving a car (i.e. turning the wheel causes the car to turn).
- *Graphs and tables*. A Bode plot of a system is a typical example of a model in a graphical form. A step response is another type of a model in its graphical form.

- *Mathematical models*. These are the classes of models that are described by differential and/or difference equations. Such models are very well suited for the analysis, prediction and design of dynamic systems, regulators and filters.

The mathematical models can be constructed in two ways:

- *Physical/Analytical modeling*. Basic laws from physics (such as Newton's laws and balance equations) are used to describe the dynamic behavior of a phenomenon or a process.
- *Empirical/Data-Based modeling*. This is an experimental approach. Some experiments are performed on the system. A model is then developed to represent the recorded data.

One can make a comparison between the physical modeling and the empirical modeling. There are many processes that are so complex in nature such that obtaining reasonable models using only physical modeling is not possible. In such situations one is forced to use the empirical techniques. [1,2,3]

A prior knowledge of the system under study is useful to make identification easier. For example, prior investigation as to whether a linear or nonlinear model is necessary. Haber [4] suggests two tests for making a decision on whether it would be worth to perform nonlinear system identification.

Generally, two approaches for 'detecting' nonlinearity have been proposed in the literature. The first is based on the notion of coherence function and higher-order spectra, the second analyzes correlations involving the residuals of a fitted linear model. [5]

Emara-Shabaik *et al* [6] have addressed the decision concerning the choice of linear versus nonlinear models using statistical hypothesis testing and the cumulants of the system output. They used the bispectrum of the output response data to differentiate among several alternative model structures in the Wiener-Hammerstein models, which are explained in Section 1.6. As an application of the tests in [6], Emara-Shabaik *et al* [7] have experimentally evaluated such tests for a laboratory-scale pH neutralization process.

In the ARMAX (AutoRegressive Moving Average with an eXogenous input) modeling, a discrete linear time-invariant system can be represented as follows:

$$A(q^{-1})y(t) = B(q^{-1})u(t) + C(q^{-1})e(t) \tag{1.1}$$

where

$u(t)$ is the n_u input vector

$y(t)$ is the n_y output vector

$e(t)$ is a white disturbance sequence

$$A(q^{-1}) = 1 + a_1 q^{-1} + \ldots + a_{n_a} q^{-n_a}$$
$$B(q^{-1}) = b_1 q^{-1} + \ldots + b_{n_b} q^{-n_b}$$
$$C(q^{-1}) = 1 + c_1 q^{-1} + \ldots + c_{n_c} q^{-n_c}$$

and q^{-1} is the backward shift operator, so $q^{-1}u(t) = u(t-1)$. The model in (1.1) can also be written as the following difference equation

$$y(t) + a_1 y(t-1) + \ldots + a_{n_a} y(t-n_a) = b_1 u(t-1) + \ldots + b_{n_b} u(t-n_b) + e(t) + c_1 e(t-1) + \ldots c_{n_c} e(t-n_c)$$

Denoting the parameters as

$$\theta = (a_1 \ldots a_{n_a}\, b_1 \ldots b_{n_b}\, c_1 \ldots c_{n_c})^{\mathrm{T}}$$

Then, equation (1.1) can be rewritten as

$$y(t) = \varphi^{T}(t)\theta + e(t)$$

where

$$\varphi(t) = (-y(t-1)...-y(t-n_a)\,u(t-1)...u(t-n_b)\,e(t-1)...e(t-n_c))^{T}$$

The elements of $\varphi(t)$ are called the regressors.

However, in practice, most processes are nonlinear and dynamic. As a result, researchers came up with different methods to represent nonlinear systems. Some well established methods that can represent wide classes of nonlinear systems are the Volterra series expansion, the Bilinear models, the NARMAX models, the Neural Networks models, and the Wiener-Hammerstein models.[1,2,3,8]

1.2 Volterra Series Expansion

Generally, any nonlinear system can be represented using the Volterra series representation as follows [3]:

$$y(t) = \sum_{n=1}^{\infty} H_n[u(t)] \tag{1.2}$$

where

$u(t)$ is the input vector

$y(t)$ is the output vector

H_n is the n-th order Volterra operator defined as

$$H_n[u(t)] = \int_{-\infty}^{\infty} \cdots \int_{-\infty}^{\infty} h_n(\tau_1,\cdots,\tau_n)u(t-\tau_1)\cdots u(t-\tau_n)d\tau_1\cdots d\tau_n \tag{1.3}$$

and

$h_n(\tau_1,\cdots,\tau_n)$ is the n-th order Volterra Kernel.

Figure 1.2 is a schematic representation of (1.2), where the total output is obtained as the sum of the outputs of the boxes with the operators $H_n[u(t)]$.

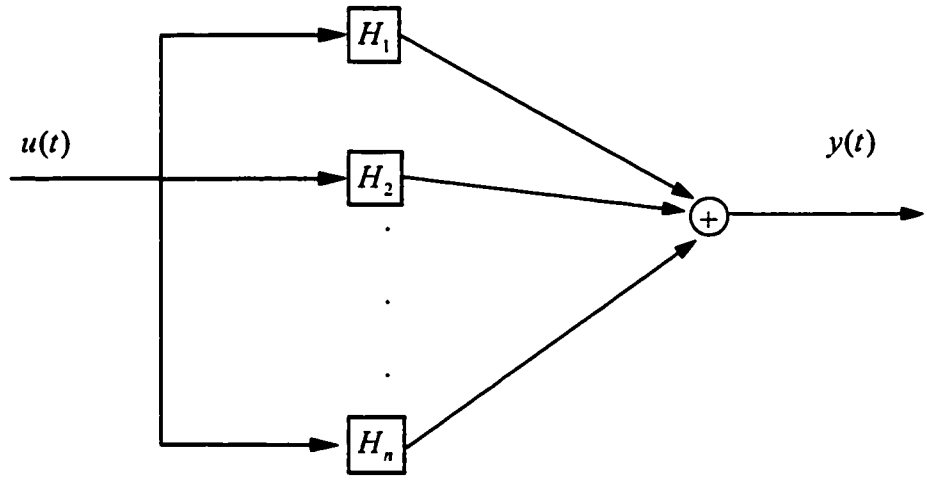


Figure 1.2. Schematic representation of a nonlinear system in terms of Volterra series.

In the above representation, $H_n[u(t)]$ is a homogeneous Volterra functional because $H_n[cu(t)] = c^n H_n[u(t)]$, in which c is a constant. Homogeneity of $H_n[u(t)]$ means that any change in the amplitude of the input results in a change of the output amplitude only but not in the output waveform. If this condition is not satisfied, the functional is called a nonhomogeneous Volterra functional. The representation of nonlinear systems using nonhomogeneous functional is called a Wiener series representation. [3,9]

While Volterra and Wiener functional series can represent a wide class of nonlinear systems, they do so at the expense of introducing an excessive number of unknown coefficients, i.e. Volterra kernels. This not only complicates the identification of

systems based on these representations, but also makes the development of controller design procedures very complex. It is perhaps for these reasons that few practical systems have been identified using Volterra and Wiener approaches and that the design of controllers based on these models have received virtually no attention.[10]

A number of ways of alleviating the difficulties of Volterra series identification have been investigated, including use of deterministic pseudo-random input or a mixture of sinusoids together with a rational-transfer-function version of this series. But nothing can nullify the essential wastefulness of the representation.[9]

1.3 Bilinear Models

As previously stated, in many cases of interest, the appropriate dynamic model may be nonlinear. In these circumstances, it is often desirable, if possible, to work with the true system description rather than a linearized approximation. To be specific, a special class is considered: discrete Bilinear models. Bilinear models comprise perhaps the simplest class of nonlinear systems. These are systems that are linear in control and linear in state but not jointly linear in state and control. In other words, they involve products of state and control. Hence the superposition principle does not apply. A single-input single-output discrete time-invariant Bilinear model can be expressed in state-space form as follows: [2]

$$\begin{aligned} x(t+1) &= Ax(t) + u(t)Nx(t) + bu(t) \\ y(t) &= cx(t) \end{aligned} \tag{1.4}$$

where $x \in \Re^n$; u, $y \in \Re$, and A, N, b and c have appropriate dimensions.

1.4 NARMAX Models

The Nonlinear AutoRegressive Moving Average with an eXogenous input (NARMAX) is a nonlinear difference equation model structure that can represent a wide class of nonlinear functions of time sequences of input-output structure. It has a general form:

$$y(t) = f(y(t-1)...y(t-n_y), u(t-d)...u(t-d-n_u), e(t-1)...e(t-n_e)) + e(t) \tag{1.5}$$

where n_u, n_y, and n_e are the dimensions of the input, output and the disturbance vectors, respectively.

The choice of the class of nonlinear function $f(.)$ is a very important task in any nonlinear system identification procedures. The polynomial NARMAX can be achieved by expanding $f(.)$ as a polynomial of a desired degree of nonlinearity, say m. For example, a NARMAX model with second order nonlinearity and without time delay can be written in general as:

$$\begin{aligned} y(t) = {} & \theta_1 y(t-1) + \theta_2 u(t-1) + \theta_3 e(t-1) + \theta_4 y^2(t-1) + \theta_5 y(t-1)u(t-1) \\ & + \theta_6 y(t-1)e(t-1) + \theta_7 u^2(t-1) + \theta_8 u(t-1)e(t-1) + \theta_9 e^2(t-1) + e(t) \end{aligned} \tag{1.6}$$

where the $\theta_i^{'}s$ are the NARMAX parameters.

The expansion $f(.)$ can represent a linear model if m is equal to one. Then model equation (1.6) will be reduced to a linear regression. However, in general, the

polynomial expansion of $f(.)$ produces a nonlinear difference equation model structure, which is still linear in the parameters. This is apparent in the model equation (1.6). [10,11]

As shown in the previous example, the nonlinearity in the NARMAX model appears in the input/output variables, namely $u(t)$ and $y(t)$. However, the NARMAX models are linear-in-parameters, which makes this type of models distinct and relatively easy to treat. On the other hand, this feature may further restrict the range of nonlinear systems that can be represented by the NARMAX models.

1.5 Artificial Neural Networks

An Artificial Neural Network (ANN) is a parallel distributed information processing structure consisting of processing elements interconnected via unidirectional signal channels. Every ANN has an input layer, an output layer and some optional layers between input and output layers, which are called hidden layers. In each layer, there are several processing elements and each processing element has a single output connection that branches into as many collateral connections as desired to the next layer. Figure 1.3 shows a feed-forward ANN architecture with just one hidden layer. The connections amongst processing elements are appropriately weighted to obtain a desired output from the ANN. There is a possibility for an additional bias term or offset for each processing element. The processing elements in each layer are called neurons. The number of neurons in the input and the output layers are determined by

the number of input and output variables in the physical system. The number of hidden layers and the number of neurons in the hidden layers are arbitrary and can vary from one to any finite number. [11,12]

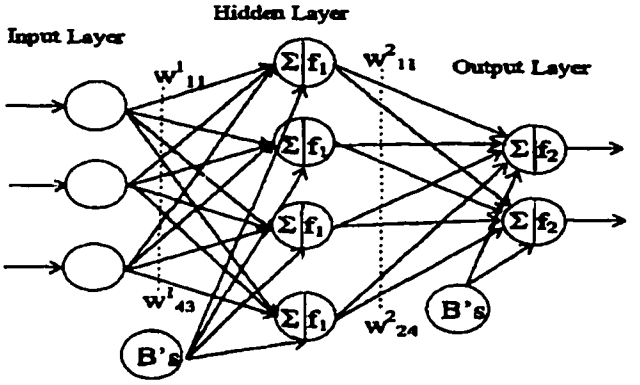


Figure 1.3. A feed-forward ANN with one hidden layer.

The processing element in each neuron can be of any mathematical type desired. For each layer, it can be a different linear or nonlinear function. However, for obtaining a nonlinear mapping, a nonlinear function is often used in the hidden layer and a linear function in the output layer for scaling purposes. For a logistic sigmoid, which is the most common nonlinear function for ANN, the output signal a_i^λ of processing element i in the layer λ can be written as:

$$a_i^\lambda = f_i^\lambda(n_i^\lambda) = \frac{1}{1+e^{-n_i^\lambda}} \tag{1.7}$$

where

$$n_i^\lambda = \sum_{j=1}^{s^{\lambda-1}} w_{ij}^\lambda f_j^{\lambda-1} + b_i^\lambda \tag{1.8}$$

and w_{ij}^{λ}'s are associated connection weights among all $s^{\lambda-1}$ neurons in layer $\lambda-1$ to neuron number i in layer λ, and b_i^{λ} is the bias for the related neurons. The Taylor's series expansion of equation (1.7) given by

$$f_i^{\lambda}(n_i^{\lambda}) = c_0 + c_1 n_i^{\lambda} + c_2 (n_i^{\lambda})^2 + c_3 (n_i^{\lambda})^3 + \ldots \tag{1.9}$$

indicates that the neuron operation can be considered as an infinite series of nonlinear expansion of its inputs.[11]

The ANN has proven capabilities of learning to approximate the nonlinear relations of the input u and the output y. The quality of learning, i.e. how well neural networks can approximate these nonlinear relations, is dependent on training algorithms. One of the most widely used training methods is back propagation learning algorithm. [11]

1.6 Wiener-Hammerstein Models

It is known that a wide class of nonlinear systems can be modeled by interconnected memoryless nonlinear gains and linear dynamic subsystems. The general class of nonlinear systems comprising static nonlinear elements and dynamic linear elements in various combinations have been studied in [4]. As shown in Figure 1.4, three basic configurations of such models are given: the Wiener model, the Hammerstein model, and the Wiener-Hammerstein model (the General model, or G-model for short). These models, also known as block-cascade models, can be used to model many practical

systems such as mechanical systems, power plants, bridges and offshore structures such as vertical piles, and oil and gas platforms. [4,6]

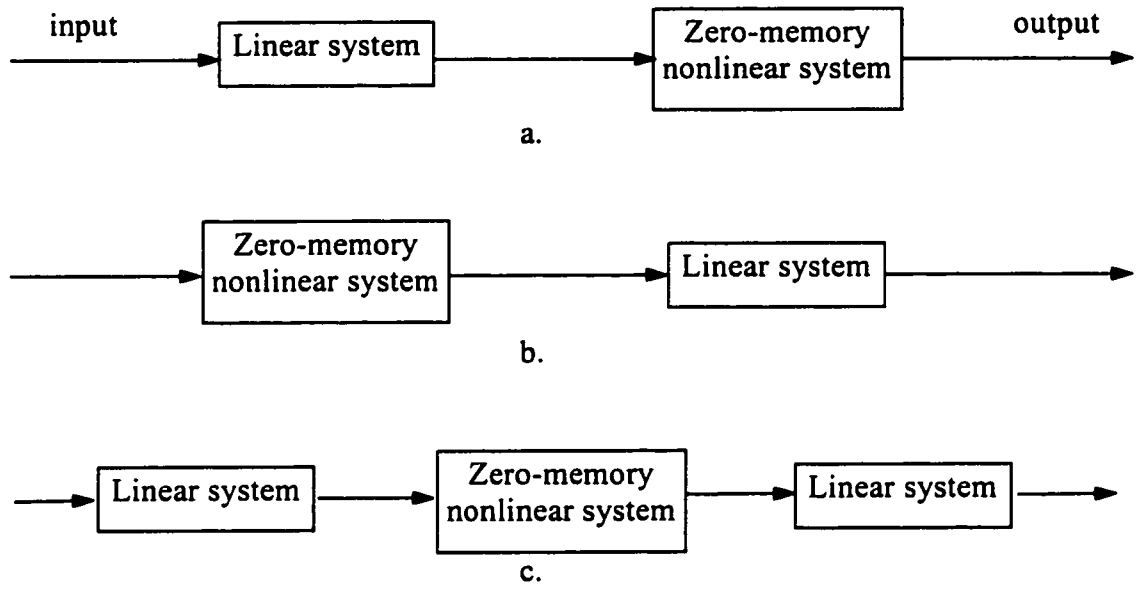


Figure 1.4. Three types of nonlinear models: a. Wiener model b. Hammerstein model c. Wiener-Hammerstein model .

The Wiener-Hammerstein model structures possess several advantages in treating nonlinear systems or processes. For many nonlinear identification problems, it has been proven that this configuration is capable of providing an accurate description of the system dynamics with finite number of parameters [6,7,12,13,14]. Also, it allows the extension of linear identification and control techniques to nonlinear systems. Moreover, in many cases the description of dynamic systems in terms of Wiener-Hammerstein models can be related to the physics of the process in hand. One of the distinct and important features of the Wiener-Hammerstein models is that the output can be written as a nonlinear function in both the parameters and the input/output

variables, which allows this type of modeling to cover a wider range of nonlinear systems than the Bilinear and NARMAX models can do.

Recently, Hsu *et al* [13] applied a Wiener-Hammerstein configuration to model the static and dynamic characteristics of the insulated gate bipolar transistor (IGBT). They used least squares methods to extract the parameters in the behavioral model from electrical measurements of physical devices. More recently, Kearney *et al* [14] showed an interesting application of the G-models in biology. They examined dynamic stiffness at the human ankle using position perturbations which were designed to provide a wide-bandwidth input and low average velocity. A Wiener-Hammerstein model was used to separate overall stiffness into intrinsic and reflux components. Intrinsic stiffness was described by a linear, second order system. Reflux stiffness dynamics were more complex, comprising a delay, a unidirectional rate-sensitive element and then lowpass dynamics.

1.6.1 Zero-memory nonlinear systems

A zero-memory nonlinear system, also known as a static nonlinear system, acts instantaneously on present inputs in some nonlinear fashion. It does not weight past inputs due to 'memory' operations as in convolution integrals for constant-parameter linear systems, where the present value of the output is a function of both present and past values of the inputs. The discussion will be restricted to such nonlinear systems where the output $y(t)$ at any time t is a single-valued nonlinear function of the input

$x(t)$ at the *same* time t. In addition, this nonlinear relation between $x(t)$ and $y(t)$ is approximated by a power polynomial of order m given by

$$y(t) = g_1 x(t) + g_2 x^2(t) + \ldots + g_m x^m(t)$$

where the coefficients g_i and the order m can be selected to approximate a given zero-memory nonlinear system. In the following, Example 1.1 illustrates how a zero-memory nonlinear function, namely a saturation function, can be approximated by a power polynomial of a finite order.

Example1.1: Approximating a saturation function by a power polynomial

Consider the following Hammerstein model

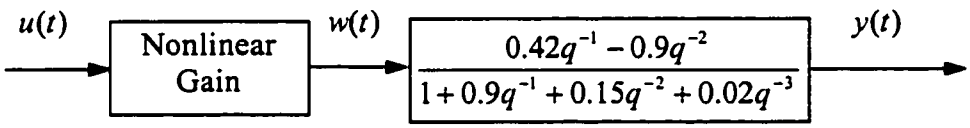


Figure 1.5. A Hammerstein model.

where q^{-1} is a backward shift operator and t is the index time.

Suppose that the nonlinear gain element in Figure 1.5 is a saturation function given by

$$w(t) = \begin{cases} 1 & u(t) > 1 \\ u(t) & |u(t)| \le 1 \\ -1 & u(t) < -1 \end{cases}$$

This nonlinear function is now approximated by the following fourth-order power polynomial

$$w(t) = g_1 u(t) + g_2 u^2(t) + g_3 u^3(t) + g_4 u^4(t)$$

The parameters g_i have no exact values to compare with. Utilizing the best fit techniques of two functions, a plot of the estimated nonlinear gain is made in Figure 1.6 for a comparison with the above saturation function.

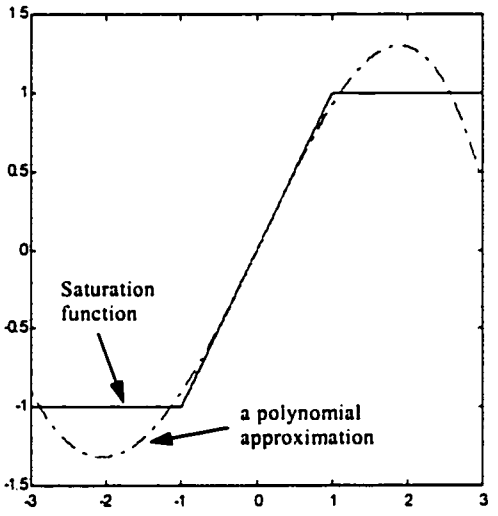


Figure 1.6. Comparison of the exact and polynomial-approximated saturation nonlinear gain.

The above approximation is somewhat acceptable within a range of $|u(t)| \leq 3.0$. So if a larger range approximation of the nonlinear gain is desired, one needs to use a higher order polynomial to approximate $w(t)$, which consequently will increase the number of operations involved in the parameters estimation process.

1.6.2 The relation between Volterra series model and the G-model

A class of nonlinear systems which consists of a linear system with impulse response $h_1(t)$ in cascade with a nonlinear zero-memory element followed by another linear system with impulse response $h_2(t)$ is shown in Figure 1.7.

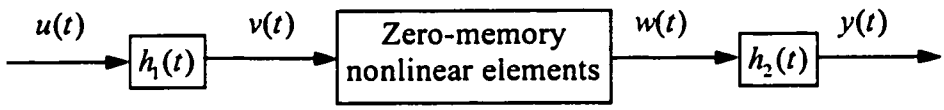


Figure 1.7. The G-model expressed in terms of the impulse responses of the linear systems.

The nonlinear zero-memory element is represented by a finite power polynomial of degree m, i.e.,

$$w(t) = g_1 v(t) + g_2 v^2(t) + \ldots + g_m v^m(t) \tag{1.10}$$

While the output $v(t)$ of the first linear system is given by the convolution summation as

$$v(t) = \sum_{\tau=-\infty}^{\infty} h_1(\tau) u(t-\tau) \ , \tag{1.11}$$

the output $w(t)$ of the nonlinear element can be expressed in terms of the input $u(t)$ in the following manner

$$\begin{aligned} w(t) = {} & g_1 \sum_{\tau_1=-\infty}^{\infty} h_1(\tau_1) u(t-\tau_1) \\ & + g_2 \sum_{\tau_1=-\infty}^{\infty} \sum_{\tau_2=-\infty}^{\infty} h_1(\tau_1) h_1(\tau_2) u(t-\tau_1) u(t-\tau_2) \\ & + \ldots \\ & + g_m \sum_{\tau_1=-\infty}^{\infty} \ldots \sum_{\tau_m=-\infty}^{\infty} h_1(\tau_1) \ldots h_1(\tau_m) u(t-\tau_1) \ldots u(t-\tau_m) \end{aligned} \tag{1.12}$$

And the final output $y(t)$ is found by the convolution summation as

$$y(t) = \sum_{\tau=-\infty}^{\infty} h_2(\tau) w(t-\tau) \tag{1.13}$$

which can be rewritten as

$$\begin{aligned} y(t) = {} & g_1 \sum_{\tau_1=-\infty}^{\infty} \sum_{\tau_2=-\infty}^{\infty} h_1(\tau_1) h_2(\tau_2) u(t-\tau_1-\tau_2) \\ & + g_2 \sum_{\tau_1=-\infty}^{\infty} \sum_{\tau_2=-\infty}^{\infty} \sum_{\tau_3=-\infty}^{\infty} h_1(\tau_1) h_1(\tau_2) h_2(\tau_3) u(t-\tau_1-\tau_3) u(t-\tau_2-\tau_2) \\ & + \ldots \\ & + g_m \sum_{\tau_1=-\infty}^{\infty} \ldots \sum_{\tau_m=-\infty}^{\infty} h_1(\tau_1) \ldots h_1(\tau_m) h_2(\tau_{m+1}) u(t-\tau_1-\tau_{m+1}) \ldots u(t-\tau_m-\tau_{m+1}) \end{aligned} \tag{1.14}$$

Therefore, for a Wiener-Hammerstein model that contains a power polynomial of order m as the nonlinear element, there is an associated Volterra series representation consisting of m Volterra kernel.[9]

Chapter 2

CLASSIFICATION OF THE ESTIMATION TECHNIQUES

2.1 Introduction

Once the nonlinear model representing the actual system has been selected, the next logical step is to carry on a certain procedure in order to identify the parameters of such a model. It is the scope of this chapter to adopt a general framework of Wiener-Hammerstein models that is going to be utilized throughout the research. In addition, the parameter estimation techniques are established for linear systems, and the

difficulties of extending such techniques to our model are highlighted. This chapter concludes by stating the thesis objectives and organization.

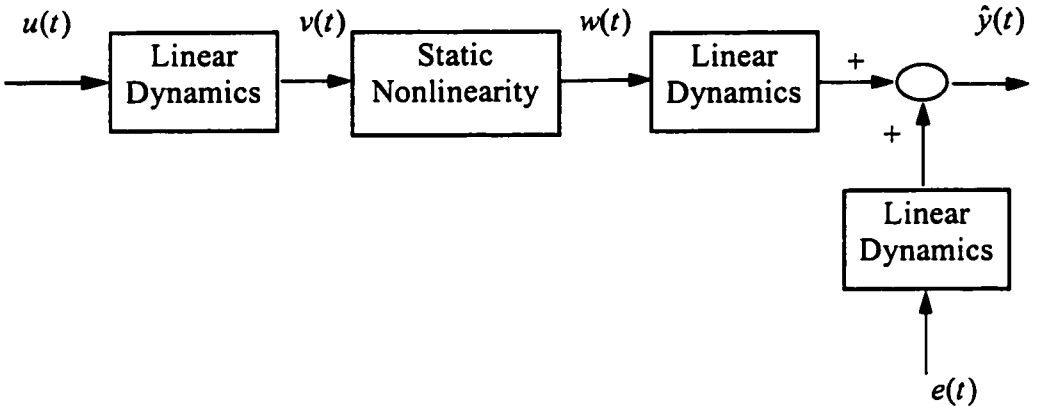


Figure 2.1. A general Wiener-Hammerstein model.

The structure of a general Wiener-Hammerstein model is depicted in Figure 2.1, where $u(t)$ is the input to the system, $\hat{y}(t)$ is the model output, $e(t)$ is a white noise sequence of zero mean and variance λ^2, and t is the time index. While $u(t)$ and $\hat{y}(t)$ are available, the intermediate variables $v(t)$ and $w(t)$ are not accessible and need be estimated.

In this thesis, the static nonlinearity is assumed to take a power polynomial model. Thus, the specific Wiener-Hammerstein model adopted in this research is shown in Figure 2.2.

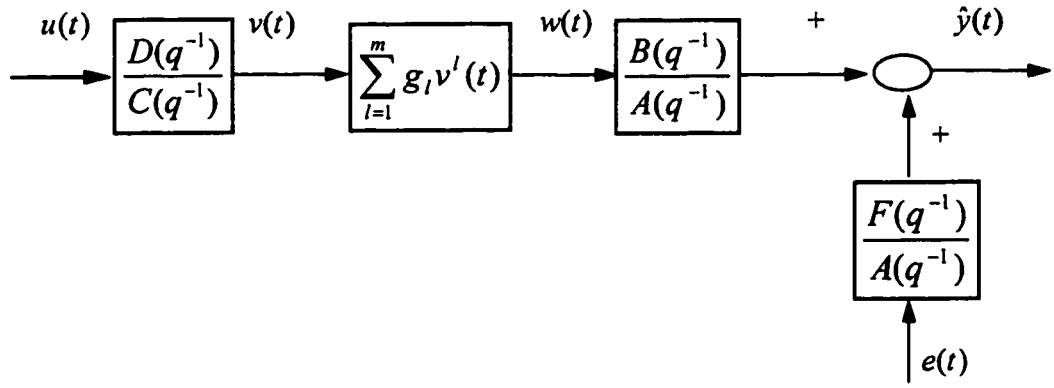


Figure 2.2. The adopted Wiener-Hammerstein model.

The output signal $\hat{y}(t)$ is given by

$$A(q^{-1})\hat{y}(t) = B(q^{-1})w(t) + F(q^{-1})e(t)$$

with

$$w(t) = g_1 v(t) + g_2 v^2(t) + \ldots + g_m v^m(t)$$

and

$$v(t) = \frac{D(q^{-1})}{C(q^{-1})}u(t)$$

The polynomial operators in the linear subsystems are defined as follows

$$A(q^{-1}) = 1 + a_1 q^{-1} + \ldots + a_{n_a} q^{-n_a}$$
$$B(q^{-1}) = 1 + b_1 q^{-1} + \ldots + b_{n_b} q^{-n_b}$$
$$C(q^{-1}) = 1 + c_1 q^{-1} + \ldots + c_{n_c} q^{-n_c}$$
$$D(q^{-1}) = 1 + d_1 q^{-1} + \ldots + d_{n_d} q^{-n_d}$$
$$F(q^{-1}) = 1 + f_1 q^{-1} + \ldots + f_{n_f} q^{-n_f}$$

q^{-1} is the shift operator. The orders m, n_a, n_b, n_c, n_d, and n_f are assumed to be known. A similar configuration has been adopted by Boutayeb and Darouach (1995).

It should be mentioned here that most of the literature treating the G-models does not include a constant term, i.e. g_0, in the power polynomial $w(t)$ [12,13].

2.2 Linear System Identification Techniques

The problem of identifying linear dynamic systems from experimental data has been studied quite well and the underlying theories are well established. The literature is full of publications that deal with such a problem in both the continuous and discrete domains, and for both offline and online approaches as well. The reader may refer to [1,2,3,8] for a comprehensive coverage of this problem. This section will only highlight the main features of the basic identification methods in linear systems for the purpose of comparing with and extending to the nonlinear model under study.

2.2.1 Nonparametric methods

The nonparameteric methods of identification are characterized by graphs or functions, which are not necessarily parameterized by a finite-dimensional parameter vector. The following are the techniques mostly used in the literature:

- *Transient analysis*. This technique is easy to apply. It gives a step response or an impulse response (weighting function) as a model. It is very sensitive to noise and can only give a crude model.
- *Frequency analysis*. The analysis herein is based on the use of sinusoids as inputs. It requires rather long identification experiments, especially if the correlation

feature is included in order to reduce sensitivity to noise. The resulting model is a frequency response. It can be presented as a Bode plot or an equivalent representation of the transfer function.

- *Correlation analysis*. The correlation analysis is based on white noise as an input. It gives the weighting function as a resulting model. It is rather insensitive to additive noise on the output signal.
- *Spectral analysis*. This analysis can be applied with rather arbitrary inputs. The transfer function is obtained in the form of a Bode plot (or other equivalent forms). To get a reasonably accurate estimate, a lag window must be used. This leads to a limited frequency resolution .[1,2]

The nonparametric methods are easy to apply but give only moderately accurate models. If high accuracy is needed, a parametric method has to be used. In such cases nonparametric methods can be used to get a first crude model, which may give useful information on how to apply the parametric methods.

2.2.2 Parametric methods

Since the field of parametric identification of linear systems is a vast and versatile area, the discussion herein is confined to the discrete linear time-invariant system given in an ARMAX form as

$$A(q^{-1})y(t) = B(q^{-1})u(t) + C(q^{-1})e(t)$$

where by taking the parameter vector as

$$\theta = (a_1 \ldots a_{n_a}\ b_1 \ldots b_{n_b}\ c_1 \ldots c_{n_c})^{\mathrm{T}}$$

the above model can be rewritten in a more convenient form

$$y(t) = \varphi^{T}(t)\theta + e(t)$$

with

$$\varphi(t) = (-y(t-1)...-y(t-n_a)u(t-1)...u(t-n_b)e(t-1)...e(t-n_c))^{T}$$

Basically, the parameter vector θ is estimated either by offline (batch) or online (recursive) procedures according to the application and/or the type of identification exercise being carried out. In the offline (batch) identification, a set of input/output data pairs are collected over a certain period of time, or sampling instants. Then, this set is used appropriately to come up with an estimate of the parameter vector θ. On the other hand, in the online schemes, the data pairs are collected and used in identification as they are generated from the plant. Once a suitable performance index (PI) is selected, the identification problem may be formulated as an optimization problem. The choice of an appropriate PI is one of the most important elements as it dictates the feasibility and effectiveness of the scheme. In general, four different approaches can be found in linear system identification literature.

- *Output error (OE) method*

The output error (OE) method is probably the most intuitively obvious approach to the problem. The OE scheme is depicted in Figure 2.3.

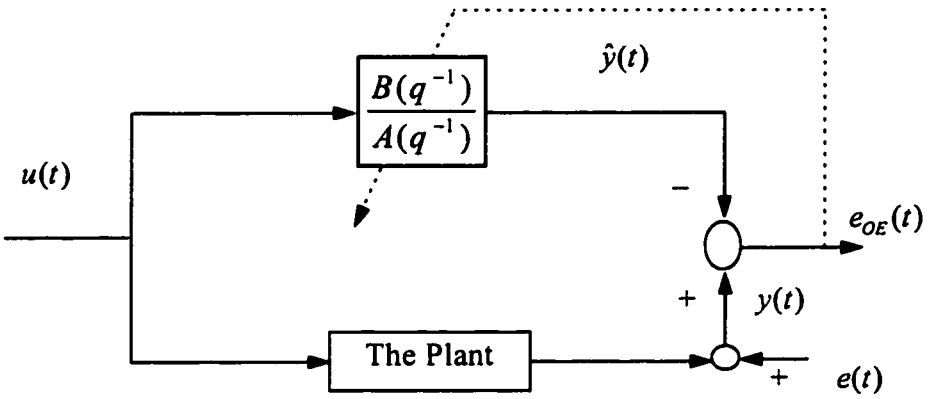


Figure 2.3. Output Error method.

As illustrated in Figure 2.3, $u(t)$ and $y(t)$ are the input/output data, $\hat{y}(t)$ is the model output, $e(t)$ is the disturbance, and the output error $e_{OE}(t)$ is defined as

$$e_{OE}(t) \equiv y(t) - \hat{y}(t)$$

The parameter vector θ is modified and updated in order to minimize the sum squared output error. A closed form expression is not available for such minimization. Rather, a numerical iterative search routine, such as Newton-Raphson algorithm, must be called. [1,2,15]

- *Least squares (LS) method*

In the least squares (LS) method, $\hat{y}(t)$ is constructed using the input as well as the past outputs. The LS scheme is displayed in Figure 2.4. The motivation stems from availability of a simple closed form expression of the parameter estimates having excellent and elegant theoretical results. However, it is well known that this scheme yields biased estimates of the parameters in the presence of noise.

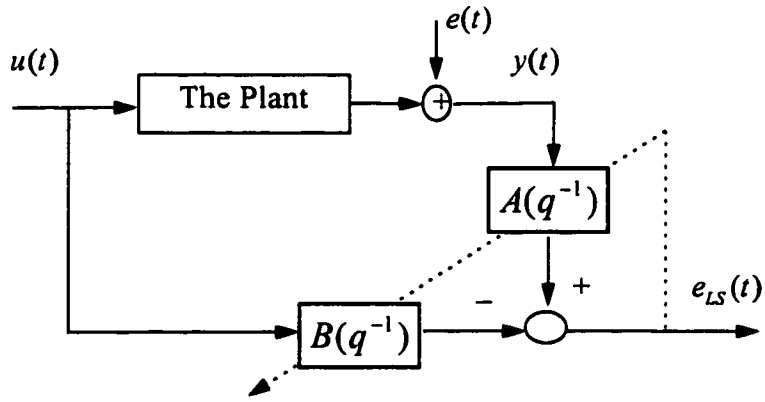


Figure 2.4. Least Squares method.

- *Prediction error (PE) method*

PE is a modification of the LS to eliminate the bias in the parameter estimates. It incorporates a noise model where the system and noise models are estimated together. Figure 2.5 may more clearly illustrate this concept. The PE minimization however requires numerical schemes.

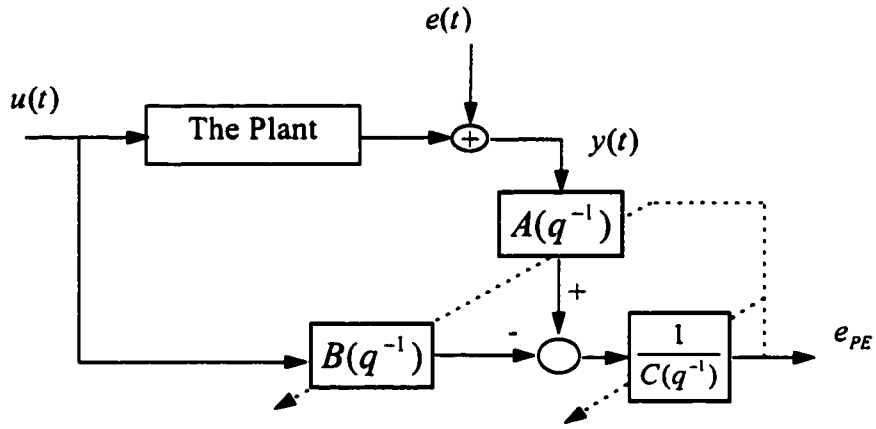


Figure 2.5. Prediction Error method.

- *Instrumental variable (IV) method*

IV is another alternative to eliminate the bias problem of the LS scheme. IV is also related to the correlation method. In this scheme, a set of variables called the instrumental variables are generated, whose entries are highly correlated to the noise-free output but are totally uncorrelated to the disturbance $e(t)$. There are many ways to generate the instrumental variables. [1,2,3,8,15]

2.3 Extending the Linear System Identification Techniques to Wiener-Hammerstein Models

The Wiener-Hammerstein model is depicted in Figure 2.2. Then the equations that govern this class of models can be written as follows.

$$C(q^{-1})v(t) = D(q^{-1})u(t) \tag{2.1}$$

$$w(t) = \sum_{l=1}^{m} g_l v^l(t) \tag{2.2}$$

$$A(q^{-1})\hat{y}(t) = B(q^{-1})w(t) + F(q^{-1})e(t) \tag{2.3}$$

By substituting (2.1) and (2.2) in equation (2.3), one would get

$$A(q^{-1})\hat{y}(t) = \sum_{l=1}^{m} g_l B(q^{-1})\left(\frac{D(q^{-1})}{C(q^{-1})}u(t)\right)^l + F(q^{-1})e(t) \tag{2.4}$$

The parameter vector that represents the model can be concatenated as follows

$$\theta = (a_1 \ldots a_{n_a} b_1 \ldots b_{n_b} c_1 \ldots c_{n_c} d_1 \ldots d_{n_d} f_1 \ldots f_{n_f} g_1 \ldots g_m)^T$$

Contrary to the Hammerstein model, the presence of polynomials C and D makes equation (2.4) nonlinear in parameters. This makes the extension of linear system identification to Wiener-Hammerstein model non-trivial. Further, one might expect a difference in performance as different linear identification techniques are extended for identification of this class of nonlinear systems.

2.4 Genetic Algorithms

The notion of Genetic Algorithms (GAs) had appeared in the literature as a technique for solving optimization problems. As will be shown in the literature survey on GAs, the GAs have been used in the identification of linear systems, as well as nonlinear systems. Here in this section, a brief introduction on the GAs is made, more details on GAs are presented in Chapter 6 of this thesis.

Genetic algorithms are directed search procedures based on the Darwinian notion of natural selection. They were first introduced by Holland, and extensively studied by Goldberg, De Jong and others [16]. One of the most successful areas of application has been the use of GAs to solve a wide variety of difficult *numerical* optimization problems. GAs are meant to complement existing optimization approaches in that they require no gradient information and are much less likely to get trapped in local minima on multi-modal surfaces. In addition, GAs have been shown to be quite insensitive to the presence of noise.

GAs maintain a population of individuals which represent potential solutions of a problem. These solutions are evaluated as to their quality (fitness), and the more fit solutions survive and produce 'offsprings' which are variations of their parents and represent solutions of potentially higher quality.

To see how these ideas are applied to function optimization, suppose without loss of generality that we want to maximize a function of n parameters $f(x_1,x_2,...,x_n)$. Each parameter is defined in a certain domain, i.e. $x_i \in D_i$ for $i = (1,2,...,n)$. Candidate solutions (individuals) in this case are just n-dimensional parameter vectors of the form:

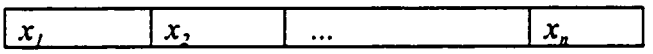


Figure 2.6. A candidate solution of the optimization problem.

which can be viewed as 'chromosomes' and the individual parameters, i.e. x_i, as 'genes'. For each such vector of parameter values, its associated function value serves as its fitness, with higher values preferred for maximization problems.

The search process begins by randomly generating an initial population of M vectors (a random sample of the parameter space) and computing the associated fitness. Individuals with higher fitness are selected to produce offsprings which inherit many but not all of the features of their parents using genetic operators like mutation and crossover. Over time, offsprings with higher fitness replace individuals with lower fitness, and both the average and maximum fitness of the population rapidly increases. [16,17]

2.5 Thesis Objectives and Organization

In this thesis, parameter estimation of Wiener-Hammerstein models will be considered. To start with, Chapter 4 introduces and discusses the properties of some typical input signals that are used in Wiener-Hammerstein models. In addition, the concept of persistence of excitation of the input signals for the proposed nonlinear type of models will be discussed. The extension of both offline and online identification techniques mentioned earlier is investigated. For the offline techniques, which are the topic of Chapter 5, the study focuses on the Output Error (OE), Least Squares (LS), Prediction Error (PE), and Instrumental Variable (IV) methods. In addition, the Evolutionary Programming techniques, presented in terms of the Genetic Algorithms (GAs), are considered as well.

For the GAs, the following essential issues are investigated and reported in Chapter 6:

- The selection of a proper fitness function in terms of suitability and computational requirements.
- Methods for generating offsprings which do not require binary coding.

The convergence and accuracy of the GAs are studied by simulation on small-scaled Wiener-Hammerstein models.

In Chapter 7, the extension and application of Recursive Prediction Error (RPE), Recursive Output Error (ROE), Recursive Least Squares (RLS), Recursive Instrumental Variable (RIV), and the Stochastic Approximation (SA) methods for the above estimation problem are investigated. A review of the available results on

conditions of convergence of the estimates are studied and included in Chapter 8. Then, these results are validated through simulation. Monte Carlo simulations are carried out to verify the statistical properties of the above estimators. The simulation results are given in details in Chapter 9.

Chapter 3

LITERATURE SURVEY

3.1 Introduction

Modeling, identification, and control design of nonlinear systems have been the subject of many research activities in the last decades. Indeed, for many dynamic systems the use of nonlinear models is often of great interest and generally characterizes physical processes adequately over their wide operating range. As a result the performance of the control law improves significantly.

The current chapter provides a literature survey on the identification of Wiener-Hammerstein models and their variations. Basically, in Section 3.2, the efforts that have been made to identify Hammerstein models are reported, while Section 3.3 addresses a literature survey on Wiener models only. Section 3.4 highlights the few publications that are concerned with the identification of the G-models. This chapter concludes by mentioning how GAs have been used in the literature of systems identification.

3.2 Hammerstein Models

One of the nonlinear realizations frequently studied is the Hammerstein model. Several classical identification techniques on this kind of models have been investigated. In system identification, one of the earliest papers on Hammerstein discrete-time models was apparently by Nareandra and Gallman [18], who estimated separately and sequentially the linear dynamic transfer function and the nonlinear static polynomial by iterative least square scheme. Since then, a number of papers have appeared that dealt with the estimation of such models. A noniterative version of this method was also proposed by Chang and Luus [19]. Kortmann and Unbehauen [20] have extended this method to the MIMO case and as well proposed a scheme based on the recursive prediction error method. Another approach based on a combination of correlation analysis and the least squares method to obtain the parameters of the Hammerstein model has been developed by Haber [21]. In [22], Stoica has presented an analytical analysis to the general convergence of an iterative

algorithm used for identification of Hammerstein systems. Recently, Rangan *et al* [23] have proposed an approach for multivariable Hammerstein identification for state-space and linear-FIR models with white noise excitation.

Ralston and Zoubir [24] have demonstrated the use of a new time-varying nonlinear model: the time-varying Hammerstein model, which is able to characterize both the time-variation and nonlinearity of the system in a simple manner.

3.3 Wiener Models

The Wiener model of a nonlinear systems is constructed by a nonlinear gain cascaded after a linear system, as shown in Figure 1.4. Using three-level-pseudorandom-sequences as input signals, Hu and Wang [25] have identified the impulse response functions of the linear part and the polynomial coefficients of the nonlinear part of the discrete-time Wiener model. Wigren [26] has derived a recursive prediction error identification algorithm to identify the Wiener model. The linear dynamic block has been modeled as a SISO transfer function operator while the static nonlinearity was approximated by a piecewise linear function. A theoretical analysis of the method has been carried out, and conditions for local convergence to the true parameter vector were given. In [27], Wigren has related the local and global convergence of his algorithm to the stability properties of an associated differential equation.

Recently, Kalafatis *et al* [28] have proposed a straightforward approach to the identification of Wiener systems in a noisy environment. The obtained models are

represented in terms of the frequency response for the linear system and the inverse of the static nonlinearity. A variety of input excitation signals, including random binary and periodic, are used in their proposed method.

3.4 Wiener-Hammerstein Models

On the other hand, very few efforts have been made to extend the methods described in the preceding sections to the general representation: the Wiener-Hammerstein model or the G-model. In [29] and [30], Billings *et al* have proposed an identification algorithm for Wiener-Hammerstein model based on correlation analysis. However, in their algorithms some restrictive assumptions are made on the input sequences in order to preserve the separability principle. In addition, the computational requirements are considerably higher.

More recently, Yoshine *et al* [31] have suggested another approach for identification of the G-model. This consists of estimating, in the SISO case, impulse responses of the linear subsystems and parameters of the nonlinear element. Boutayeb and Darouach [32] have presented a recursive method to estimate parameters of the linear and nonlinear parts of the G-model by applying the Weighted Extended RLS. In addition, they have extended their results to the MISO Wiener-Hammerstein models and established conditions for parameter convergence to the true values.

The crucial step in the identification of Wiener-Hammerstein nonlinear models is the selection of the model structure: i.e. the orders m, n_a, n_b, n_c, n_d, and n_f of the

linear subsystems and the zero-memory nonlinearity shown in Figure 2.2. Many approaches have been developed to resolve this problem. A survey of these attempts is given by Haber and Unbehauen [4]. More recently, Emara-Shabaik and Moustafa [33] have developed a structure identification criterion based on the bispectrum and bicoherence of the output sequence only for the class of Wiener-Hammerstein models.

3.5 Genetic Algorithms in Systems Identification

The underlying principles of GAs were first published by Holland in 1962 [34]. The mathematical framework was developed in the late 1960's. Caponetto *et al* [35] gave an overview on GAs, describing their main operative features and relative implementation problems. They provided a number of applications in systems and electrical engineering where GAs are implemented. In the last few years, research devoted to GAs has significantly increased. A complete reference on GAs and how they are implemented can be found in the book by Goldberg [36].

The work by Kristinsson *et al* [17,37] can be considered as one of the earliest efforts in using GAs in systems identification and control. They have implemented a GA as an estimator for discrete-time and continuos-time linear systems. The results they obtained employing this algorithm are considered to be well suited to the adaptive control problem in combination with a pole placement scheme that utilizes that knowledge of poles and zeros. Sheta *et al* [38] continued to explore the use of GAs for solving parameter estimation problems in nonlinear systems. They have

considered systems operating in noisy environments whose initial conditions can vary widely. Billings *et al* [39] have presented a new rational model identification algorithm based on GAs. They have shown, by simulation, that the results obtained using the GAs were achieved at the expense of excessive computations.

Etter *et al* [40] have presented an adaptive genetic algorithm for determining the optimum coefficients in a recursive adaptive linear filter. They used their algorithm in IIR filter adaptations with cases of a unimodal error and a bimodal error surface. They have not proven convergence of the algorithm to the global minimum. But they demonstrated the ability of this algorithm to adaptively distinguish the global minimum from local minima. A similar work was proposed by Grefenstette [41]. He attempted to determine the optimal control parameters using GAs. A more recent work has been reported by Lansberry *et al* [42]. They investigated the application of a GA for optimizing the gains of a proportional-plus-integral controller for a hydrogenerator plant. They demonstrated the successful use of a GA on a digital computer with plant modeling performed on an analog machine. The results they obtained indicated that the GAs can be successfully applied in a noisy analog environment with sufficient speed to optimize hydrogenerator governors in real-time.

Chapter 4

INPUT SIGNALS AND PERSISTENCE OF EXCITATION

4.1 Introduction

Linear system models often fail to describe the complete dynamics of actual physical systems. Nonlinear models overcome many of the limitations of linear system models. In order to uniquely identify the parameters of a system model, the input used in the identification experiment must be sufficiently 'rich'. The choice of the input signal plays an important role in obtaining accurate parameter estimates from the

identification experiment. In other words, the input signal used in an identification experiment can have a significant influence on the resulting parameter estimates [1].

Basically, this chapter presents an analysis of random as well as pseudo-random signals and their variants. The spectral characteristics of such signals are employed in the analysis. Furthermore, based on the pseudo-random signals, the persistence of excitation concept, which is well established and understood for linear system identification, is extended to the G-models via the Volterra series representation of nonlinear systems.

4.2 Random Signals

A random signal, or sequence, is termed as *colored* if its values are correlated in time, which means that the prediction of a future value, or a set of values, of such a signal changes with the knowledge of its previous and present values. On the other hand, a *white* signal has its values, at different instants of time, completely uncorrelated. A typical excitation random signal for system identification is the Gaussian White Noise (GWN) with certain mean and variance. The cross-correlation identification methods of nonlinear systems are based on GWN excitation. Another widely used random sequence is the Uniform White Noise (UWN) with a specific mean.[44,45]

The relation between the white and colored signals is depicted in Figure 4.1. The colored noise can generally be modeled as an output of a linear dynamical system which is driven by a white noise sequence.

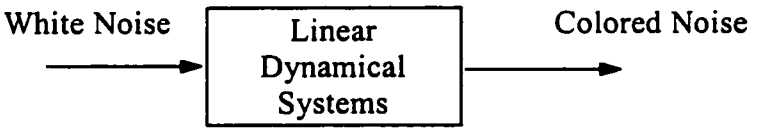


Figure 4.1. Relation between white and colored noise signals.

The power spectrum of each of the two random signals is as shown in Figure 4.2. The white noise has a constant power for all frequencies, while the colored noise has a frequency dependent power.

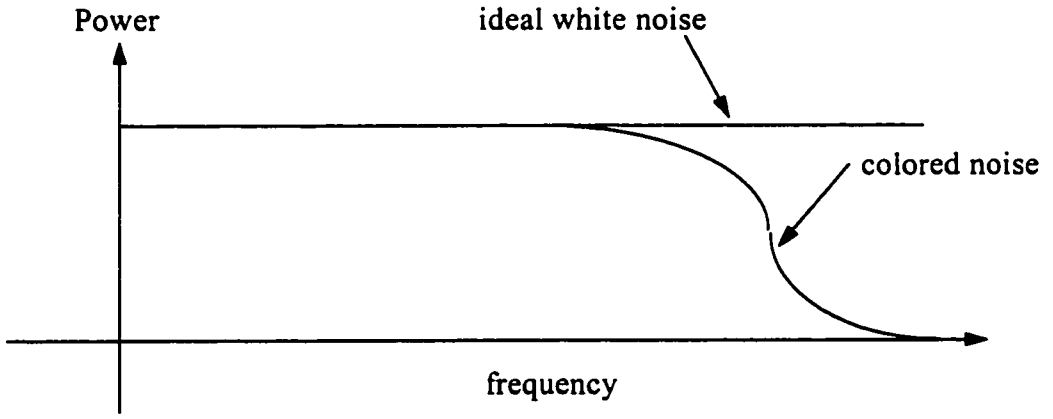


Figure 4.2. Power spectrum of both white and colored noise signals.

The characteristic of having a constant power spectrum enables the white noise to excite all the modes of the system being identified. Hence, it makes the white noise signal an ideal input for system identification experiments. However, it is the implementation and the practicality of such a signal that limit, and reduce, its use in the real-world system identification problems. This limitation motivates the need for

other types of input signals that have almost similar characteristics to those of the white noise.[43]

4.3 Pseudo-Random Signals

The pseudo-random (PR) signals are periodic signals which have many similar time- and frequency- domain properties of purely random signals. But, because they are deterministic, experiments can be repeated. Due to their implementation simplicity, PR input signals have for many years been employed in a wide range of applications, and particularly as input stimuli for system identification.[44,45]

4.3.1 Pseudo-random binary signals (PRBS)

In the 1960's and early 1970's, there was a substantial research in the design and application of PR signals. Pseudo-Random Binary Signals (PRBS) based on maximum-length sequences are easy to generate using simple shift register circuitry with appropriate feedback. This has resulted in their incorporation as a routine facility in a number of signal generators and their use in a wide range of linear system identification applications. Figure 4.3 shows a typical PRBS.[1,45,46]

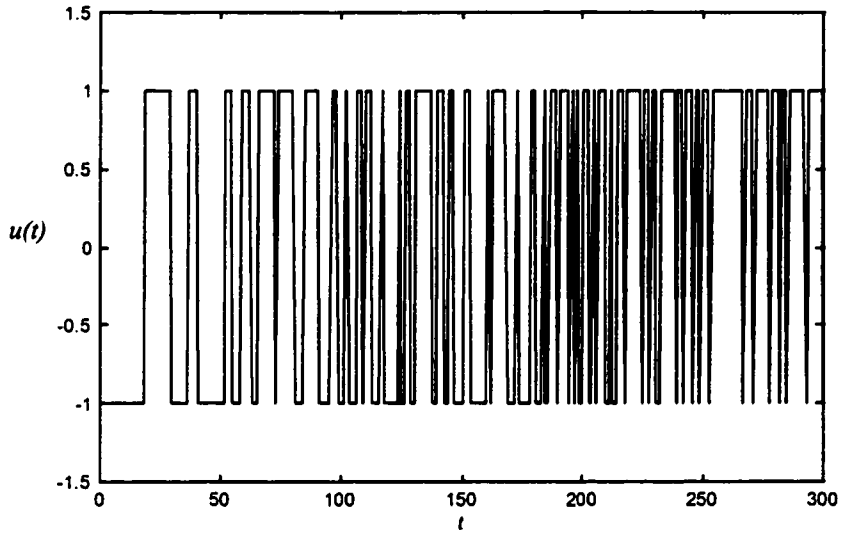


Figure 4.3. A typical PRBS.

The interest in pseudo-random signals resulted from their good autocorrelation properties (close to that of a white noise). Provided that the clock pulse interval is short compared with the dynamics of the system being identified, PRBS can be regarded as having an autocorrelation function approximating that of a white noise signal. In the frequency domain, this corresponds to the white noise signal having a flat power spectrum over the bandwidth of the system being identified. Figure 4.4 represents a simulated normalized autocorrelation function for the PRBS shown in Figure 4.3.[44]

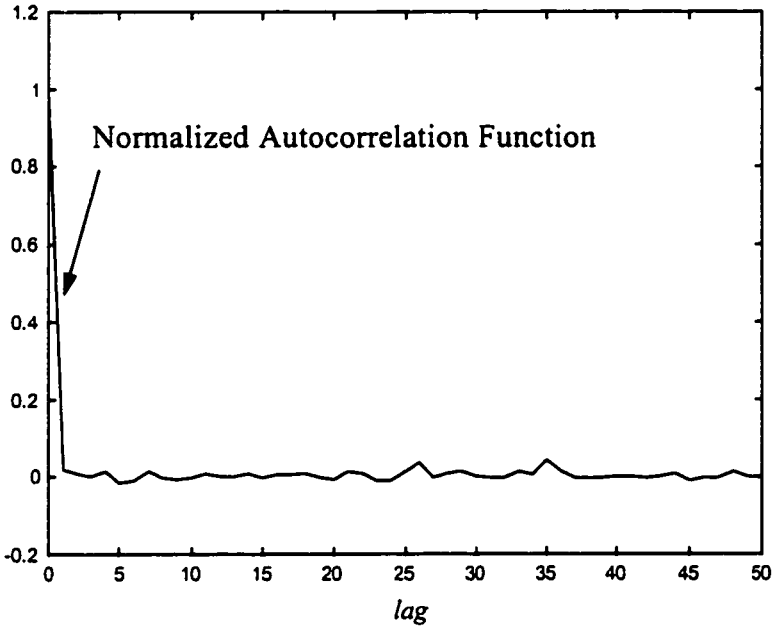


Figure 4.4. Normalized autocorrelation function for the above PRBS.

4.3.2 Pseudo-random multilevel signals (PRMS)

The other type of pseudo-random signals is the maximum-length Pseudo-Random Multilevel Signals (PRMS). The PRMS have desirable properties in the identification of nonlinear systems. Their time and frequency domain properties are more similar to the ones of the white noise than the PRBS.[44,45,46]

In the rest of this subsection, we introduce an excitation signal that mimics the PRMS in the sense of having multiple levels to switch between, while it is easier to generate and implement. Based on a random number generator, this modified PRMS (MPRMS) switches, with equal probability, between q deterministic levels at random switching times. The switching time is further restricted by a low and a high number of samples. The signal will randomly switch between any two levels at some random

time instant between t_l and t_h. Figures 4.5 and 4.6 show a typical 4-level MPRMS and its normalized autocorrelation function.

The use of the MPRMS has several advantages over the PRMS. Among which is that while the q-level PRMS is generated using a q-level feedback shift-register generator, the MPRMS can be easily generated by the means of random number generator. In addition, the PRMS generation and properties depend heavily on the algebra of finite fields, which is not easy to generate as reported in [46]. Whereas the MPRMS is intuitive and simple to generate and implement.

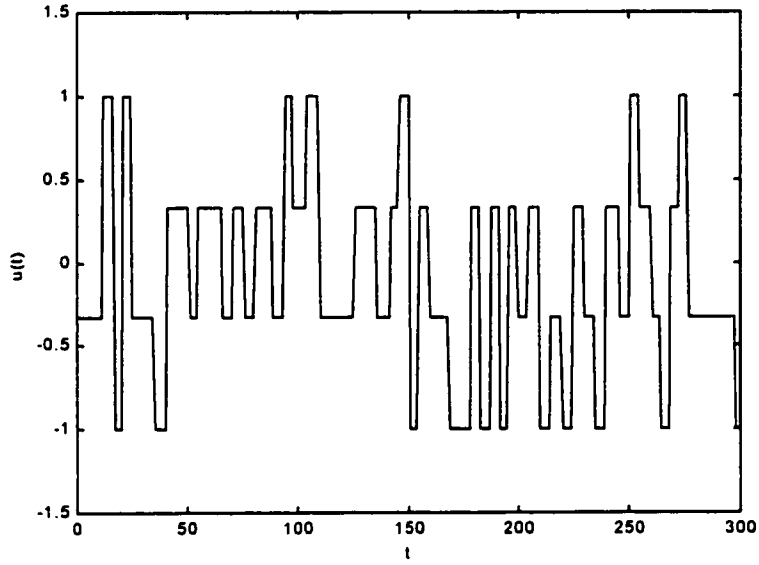


Figure 4.5. A 4-level MPRMS.

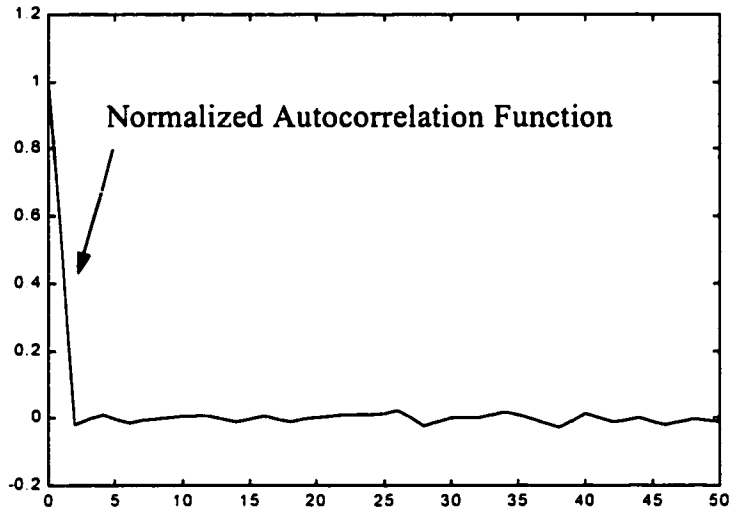


Figure 4.6. Normalized autocorrelation function for the above 4-level MPRMS.

4.4 Persistence of Excitation for Nonlinear Systems

For linear systems, a signal $u(t)$ is said to be persistently exciting (pe) of order n if:

(i) the following limit exists:

$$r_u(\tau) = \lim_{N\to\infty} \frac{1}{N} \sum_{t=1}^{N} u(t+\tau)u^{\mathrm{T}}(t); \quad \tau = 0,1,2,\cdots$$

and

(ii) the matrix

$$R_u(n) = \begin{pmatrix} r_u(0) & r_u(1) & \cdots & r_u(n-1) \\ r_u(-1) & r_u(0) & & \vdots \\ \vdots & & \ddots & \vdots \\ r_u(1-n) & \cdots & \cdots & r_u(0) \end{pmatrix}$$

is positive definite.[1,2,8]

However, for nonlinear system identification, there is a lack of a common definition of persistency of excitation conditions. Unfortunately, with finite length GWN data records, the cross-correlation identification method only gives approximate estimates of the true system's Volterra kernels and the estimation error increases as record length decreases. PRBS have been used extensively for linear system identification. However, PRBS do not sufficiently excite nonlinear systems. In fact, even quadratic Volterra systems cannot be completely identified using PRBS[48]. Glad *et al* [48] have defined a generalized version of the concept of persistence of excitation described above. They related it to the global identifiability of a general model that is described by a set of parameterized differential algebraic equations. However, they did not provide a unique or exact definition of persistence of excitation for nonlinear system.

Perhaps one of the trials that aimed to establish a sort of a qualification of an input signal as being pe and relatively relevant to the research herein was proposed by Nowak *et al* [48]. They considered a pseudo-random multilevel signal (PRMS) for the identification of nonlinear systems modeled via a truncated Volterra series with a finite degree of nonlinearity and finite memory length. It was shown that PRMS's are pe for a Volterra series model with nonlinearities of polynomial degree N if and only if the input sequence takes on $N+1$ or more distinct levels.

Based on this condition and referring to section 1.6.2, where the relation between Volterra and G-models was established, it may be claimed that a PRMS is rich

enough for a G-model with a nonlinear power polynomial of order q if it is constructed from at least $q+1$ distinct levels.

Chapter 5

OFFLINE IDENTIFICATION OF WIENER-HAMMERSTEIN MODELS

5.1 Introduction

Offline (batch) algorithms process all the input/output data simultaneously and produce a single estimate of the parameter vector. On contrast, the online (recursive) methods of Chapter 7 process the observations at one sampling instant at a time and update the parameter estimates each time. Offline algorithms are suitable only when estimates are required once or at long intervals, or when adequate computation is

available, since they process the entire record every time. In this chapter, the offline methods of the linear system models are extended to the Wiener-Hammerstein models. Namely, Output Error (OE), Prediction Error (PE), Nonlinear Least Squares (NLS), and Instrumental Variable (IV) are considered. The motivations for these methods as well are included.

5.2 Output Error (OE) Method

As stated in Chapter 2, the OE method is the most intuitive identification scheme. It attempts to match the model output to the actual observations by adjusting the model parameters. The OE method has a wide range of applications, among which is the Model Reference Adaptive Systems (MRAS); see Astrom, Chapter 4 [49]. In the sequel, we derive the Output Error (OE) for Wiener-Hammerstein models.

The OE method is schematically depicted in Figure 5.1.

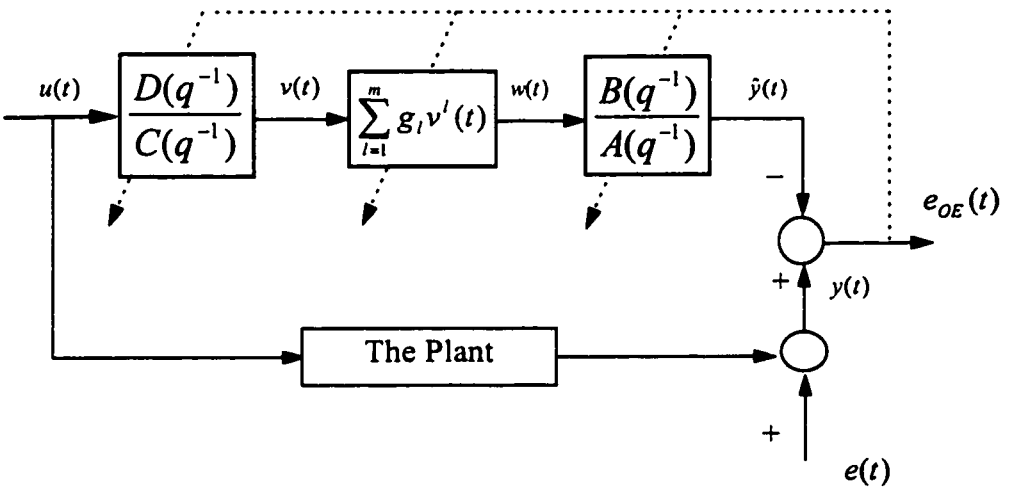


Figure 5.1. Output Error method for the Wiener-Hammerstein model.

where $v(t), w(t)$, and $\hat{y}(t)$ are the intermediate variables and the predicted output, respectively. These variables are defined as follows

$$\begin{aligned} C(q^{-1})v(t) &= D(q^{-1})u(t) \\ w(t) &= \sum_{l=1}^{m} g_l v^l(t) \\ A(q^{-1})\hat{y}(t) &= B(q^{-1})w(t) \end{aligned} \tag{5.1}$$

Further, the parameter vector is defined as

$$\theta = (a_1 \ldots a_{n_a} b_1 \ldots b_{n_b} c_1 \ldots c_{n_c} d_1 \ldots d_{n_d} g_1 \ldots g_m)^T \tag{5.2}$$

Now, define the output error $e_{OE}(t,\theta)$ as

$$e_{OE}(t,\theta) \equiv y(t) - \hat{y}(t) \tag{5.3}$$

The output error estimation of the parameter vector θ is obtained by minimizing the following criterion function

$$V_N(\theta) = \frac{1}{N}\sum_{t=1}^{N} e_{OE}^2(t,\theta) \tag{5.4}$$

where N is the total number of observations. Next, introduce the notation

$$\psi_{OE}(t,\theta) = -\left(\frac{\partial e_{OE}(t,\theta)}{\partial \theta}\right)^{\mathrm{T}} = \left(\frac{\partial \hat{y}(t,\theta)}{\partial \theta}\right)^{\mathrm{T}} \tag{5.5}$$

which is a row vector. The estimate $\hat{\theta}$ is obtained by setting the derivatives of $V_N(\theta)$ with respect to θ, denoted by $V_N'(\theta)$, equal to zero. This gives

$$V_N'(\theta) = -\frac{2}{N}\sum_{t=1}^{N} e_{OE}(t,\theta)\psi_{OE}^{\mathrm{T}}(t,\theta) = \mathbf{0} \tag{5.6}$$

The elements of $\psi_{OE}(t,\theta)$ are obtained using equations (5.1), (5.3), and (5.5) as follows:

$$A(q^{-1})\frac{\partial \hat{y}(t)}{\partial a_i} = -\hat{y}(t-i) \tag{5.7}$$

$$A(q^{-1})\frac{\partial \hat{y}(t)}{\partial b_i} = w(t-i) \tag{5.8}$$

$$\begin{aligned} A(q^{-1})\frac{\partial \hat{y}(t)}{\partial c_i} &= B(q^{-1})\frac{\partial w(t)}{\partial c_i} \\ \frac{\partial w(t)}{\partial c_i} &= \sum_{l=1}^{m} l \times g_l v^{l-1}(t)\frac{\partial v(t)}{\partial c_i} \\ C(q^{-1})\frac{\partial v(t)}{\partial c_i} &= -v(t-i) \end{aligned} \tag{5.9}$$

$$\begin{aligned} A(q^{-1})\frac{\partial \hat{y}(t)}{\partial d_i} &= B(q^{-1})\frac{\partial w(t)}{\partial d_i} \\ \frac{\partial w(t)}{\partial d_i} &= \sum_{l=1}^{m} l \times g_l v^{l-1}(t)\frac{\partial v(t)}{\partial d_i} \\ C(q^{-1})\frac{\partial v(t)}{\partial d_i} &= u(t-i) \end{aligned} \tag{5.10}$$

$$A(q^{-1})\frac{\partial \hat{y}(t)}{\partial g_i} = B(q^{-1})\frac{\partial w(t)}{\partial g_i}$$
$$\frac{\partial \hat{w}(t)}{\partial g_i} = v^i(t) \qquad (5.11)$$

5.3 A Nonlinear Least Squares (NLS) Method

The Least Squares (LS) method is a special case of the general PE described in the next section. While in the PE method the disturbance $e(t)$ is modeled and minimized, via minimization of a white sequence, the LS method is derived without taking the disturbance statistics into consideration. That is why the LS, for linear system models, works better when there is no disturbance or the so-called equation error, i.e. e_{LS}, is white. Since the output of a Wiener-Hammerstein model is not linear in the parameter vector θ, an exact least squares identification method cannot be derived. In the following we present a nonlinear least squares algorithm that is based on a similar principle. The algorithm is schematically depicted in Figure 5.2.

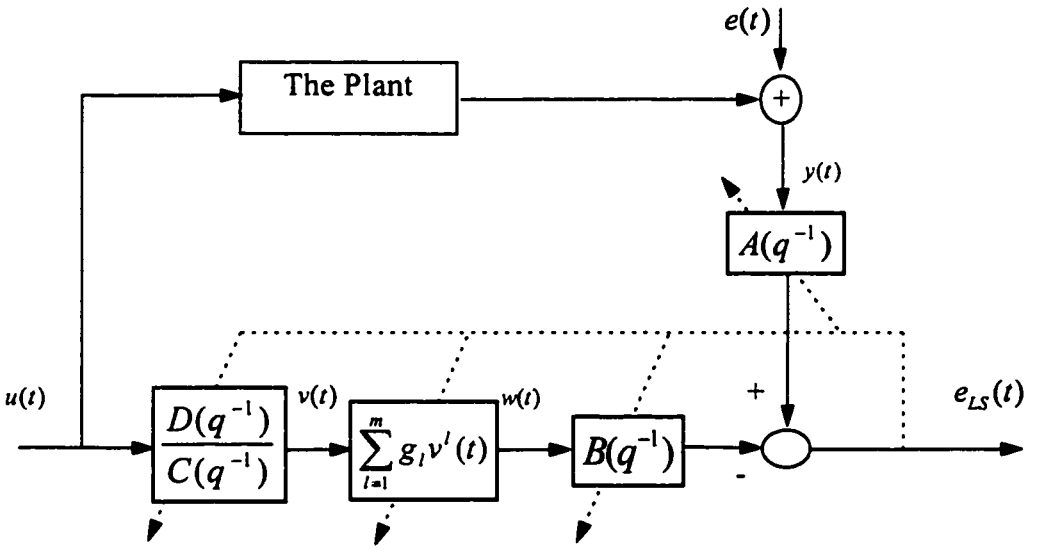


Figure 5.2. Least Squares method for the Wiener-Hammerstein model.

The intermediate variables $v(t)$ and $w(t)$ are computed as follows:

$$\begin{aligned} C(q^{-1})v(t) &= D(q^{-1})u(t) \\ w(t) &= \sum_{l=1}^{m} g_l v^l(t) \end{aligned} \tag{5.12}$$

As earlier, the parameter vector is defined as

$$\theta = (a_1 \ldots a_{n_a} b_1 \ldots b_{n_b} c_1 \ldots c_{n_c} d_1 \ldots d_{n_d} g_1 \ldots g_m)^{\mathrm{T}} \tag{5.13}$$

Now, define the output error $e_{LS}(t,\theta)$ as

$$e_{LS}(t,\theta) \equiv A(q^{-1})y(t) - B(q^{-1})w(t) \tag{5.14}$$

and the criterion function as

$$V_N(\theta) = \frac{1}{N}\sum_{t=1}^{N} e_{LS}^2(t,\theta) \tag{5.15}$$

The estimates $\hat{\theta}$ of the vector θ is obtained by setting the derivatives of $V_N(\theta)$, with respect to θ, equal to zero, i.e.

$$V_N'(\theta) = -\frac{2}{N}\sum_{t=1}^{N} e_{LS}(t,\theta)\psi_{LS}^{T}(t,\theta) = \mathbf{0} \tag{5.16}$$

where

$$\psi_{LS}(t,\theta) = -\left(\frac{\partial e_{LS}(t,\theta)}{\partial \theta}\right)^{T} \tag{5.17}$$

Here $\psi_{LS}(t,\theta)$ is a row vector whose elements are obtained using equation (5.14), (5.16), and (5.17) as follows:

$$\frac{\partial e_{LS}(t)}{\partial a_i} = y(t-i) \tag{5.18}$$

$$\frac{\partial e_{LS}(t)}{\partial b_i} = -w(t-i) \tag{5.19}$$

$$\begin{aligned} &\frac{\partial e_{LS}(t)}{\partial c_i} = -B(q^{-1})\frac{\partial w(t)}{\partial c_i} \\ &\frac{\partial w(t)}{\partial c_i} = \sum_{l=1}^{m} l \times g_l v^{l-1}(t)\frac{\partial v(t)}{\partial c_i} \\ &C(q^{-1})\frac{\partial v(t)}{\partial c_i} = -v(t-i) \end{aligned} \tag{5.20}$$

$$\begin{aligned} &\frac{\partial e_{LS}(t)}{\partial d_i} = -B(q^{-1})\frac{\partial w(t)}{\partial d_i} \\ &\frac{\partial w(t)}{\partial d_i} = \sum_{l=1}^{m} l \times g_l v^{l-1}(t)\frac{\partial v(t)}{\partial d_i} \\ &C(q^{-1})\frac{\partial v(t)}{\partial d_i} = u(t-i) \end{aligned} \tag{5.21}$$

$$\begin{aligned} &\frac{\partial e_{LS}(t)}{\partial g_i} = -B(q^{-1})\frac{\partial w(t)}{\partial g_i} \\ &\frac{\partial w(t)}{\partial g_i} = v^{i}(t) \end{aligned} \tag{5.22}$$

It will be shown through simulation in Chapter 9 that generally this algorithm provides biased parameter estimates in the presence of noise. This result, however, is expected from the analysis in the linear system identification.

5.4 Prediction Error (PE) Method

The Prediction Error (PE) is a modification of NLS in order to eliminate the bias problem. The bias is eliminated by incorporating a noise model where the system as well as the noise parameters are estimated simultaneously. The scheme discussed herein is based on a similar principle as that of the linear system models. However, because $v(t)$ and $w(t)$ are not accessible, they must be computed using the most recent estimates of $C(q^{-1})$, $D(q^{-1})$ and $g_i's$. The PE identification scheme is shown in Figure 5.3.

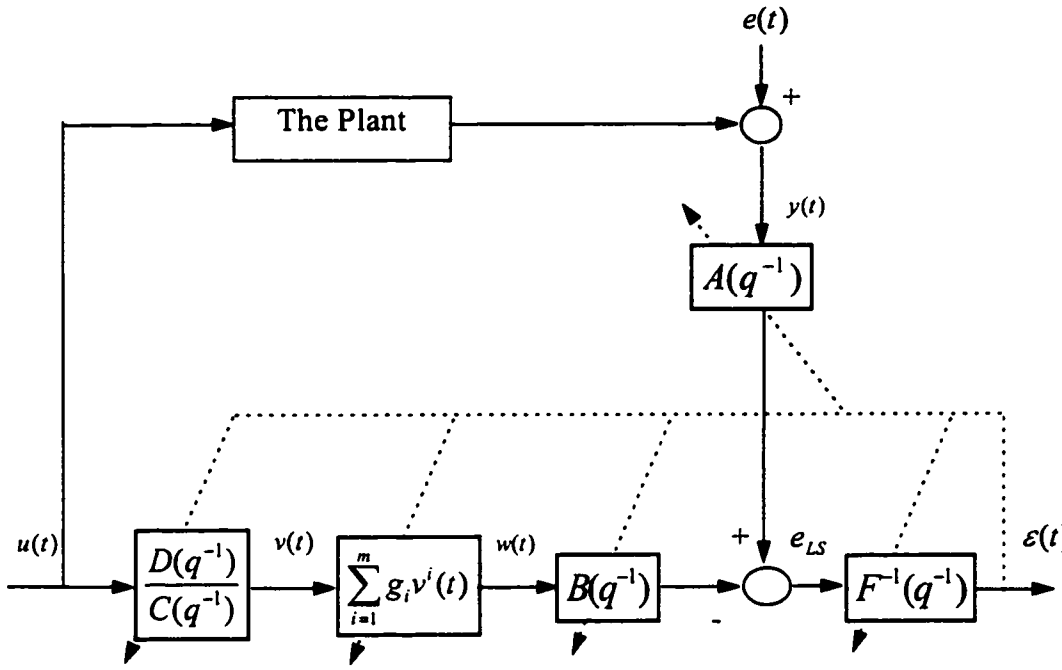


Figure 5.3. Prediction Error method for the Wiener-Hammerstein model

In the above $v(t)$ and $w(t)$ are the two intermediate variables computed as follows

$$C(q^{-1})v(t) = D(q^{-1})u(t)$$
$$w(t) = \sum_{i=1}^{m} g_i v^i(t) \tag{5.23}$$

In the PE method, the bias problem is resolved by assuming an AutoRegressive (AR) model for e_{LS} and minimizing the estimated variance of a white sequence, i.e.

$$e_{LS}(t) = F(q^{-1})\varepsilon(t)$$

where $\varepsilon(t)$ is assumed white, and adequate terms are considered in $F(q^{-1})$ to validate the above.

The prediction error $\varepsilon(t)$ now can be computed as follows

$$A(q^{-1})y(t) = B(q^{-1})w(t) + F(q^{-1})\varepsilon(t) \tag{5.24}$$

The parameter vector for the PE method is defined as:

$$\theta = (a_1 \ldots a_{n_a} b_1 \ldots b_{n_b} c_1 \ldots c_{n_c} d_1 \ldots d_{n_d} f_1 \ldots f_{n_f} g_1 \ldots g_m)^{\mathrm{T}} \tag{5.25}$$

Equation (5.24) can now be rewritten as:

$$\varepsilon(t) = A(q^{-1})y(t) - B(q^{-1})w(t) - \overline{F}(q^{-1})\varepsilon(t) \tag{5.26}$$

where

$$\overline{F}(q^{-1}) = [F(q^{-1}) - 1]$$

More formally, the prediction error $\varepsilon(t,\theta)$ is defined as

$$\varepsilon(t,\theta) = A(q^{-1})y(t) - B(q^{-1})w(t) - [F(q^{-1}) - 1]\varepsilon(t,\theta) \tag{5.27}$$

In order to obtain an estimate of the parameter vector θ via PE method, the following criterion function can be used:

$$V_N(\theta) = \frac{1}{N}\sum_{t=1}^{N} \varepsilon^2(t,\theta) \tag{5.28}$$

Now, $\hat{\theta}$ is obtained by setting the derivatives of $V_N(\theta)$, with respect to θ, equal to zero

$$V_N'(\theta) = -\frac{2}{N}\sum_{t=1}^{N} \varepsilon(t,\theta)\psi_{PE}^{\mathrm{T}}(t,\theta) = \mathbf{0} \tag{5.29}$$

where

$$\psi_{PE}(t,\theta) = -\left(\frac{\partial \varepsilon(t,\theta)}{\partial \theta}\right)^{\mathrm{T}} \tag{5.30}$$

Again, $\psi_{PE}(t,\theta)$ is a row vector. The elements of $\psi_{PE}(t,\theta)$ are obtained using equations (5.27), (5.29), and (5.30) as

$$F(q^{-1})\frac{\partial \varepsilon(t)}{\partial a_i} = y(t-i) \tag{5.31}$$

$$F(q^{-1})\frac{\partial \varepsilon(t)}{\partial b_i} = -w(t-i) \tag{5.32}$$

$$\begin{aligned} F(q^{-1})\frac{\partial \varepsilon(t)}{\partial c_i} &= -B(q^{-1})\frac{\partial w(t)}{\partial c_i} \\ \frac{\partial w(t)}{\partial c_i} &= \sum_{l=1}^{m} l \times g_l v^{l-1}(t)\frac{\partial v(t)}{\partial c_i} \\ C(q^{-1})\frac{\partial v(t)}{\partial c_i} &= -v(t-i) \end{aligned} \tag{5.33}$$

$$\begin{aligned} F(q^{-1})\frac{\partial \varepsilon(t)}{\partial d_i} &= -B(q^{-1})\frac{\partial w(t)}{\partial d_i} \\ \frac{\partial w(t)}{\partial d_i} &= \sum_{l=1}^{m} l \times g_l v^{l-1}(t)\frac{\partial v(t)}{\partial d_i} \\ C(q^{-1})\frac{\partial v(t)}{\partial d_i} &= u(t-i) \end{aligned} \tag{5.34}$$

$$F(q^{-1})\frac{\partial \varepsilon(t)}{\partial f_i} = -\varepsilon(t-i) \tag{5.35}$$

$$\begin{aligned} F(q^{-1})\frac{\partial \varepsilon(t)}{\partial g_i} &= -B(q^{-1})\frac{\partial w(t)}{\partial g_i} \\ \frac{\partial w(t)}{\partial g_i} &= v^i(t) \end{aligned} \tag{5.36}$$

In all the above algorithms, namely the OE, PE, and NLS, the normal equations are nonlinear in the parameters. Therefore, there is no closed form, i.e. analytical, solution for such equations. However, the parameter vector estimates can be obtained by means of suitable numerical optimization techniques, which will be discussed next.

5.5 Optimization Algorithms

Since the minimum of $V_N(\theta)$ cannot be found analytically in the above offline algorithms, the minimization must be performed using a numerical search routine. A commonly used method is the *Newton-Raphson algorithm*:

$$\hat{\theta}^{(k+1)} = \hat{\theta}^{(k)} - \alpha_k \left[V_N''(\hat{\theta}^{(k)})\right]^{-1} V_N'(\hat{\theta}^{(k)}) \tag{5.37}$$

Here $\hat{\theta}^{(k)}$ denote the kth iteration parameter estimate vector in the search. The sequence of scalars α_k in (5.37) is used to control the step length. In the following, we specifically consider the Prediction Error method. The derivation in other schemes follows a similar approach.

As shown in equation (5.29) the derivative of $V_N(\theta)$ can be obtained as:

$$V_N'(\theta) = -\frac{2}{N}\sum_{t=1}^{N} \varepsilon(t,\theta)\psi_{PE}^{T}(t,\theta) \tag{5.38}$$

Now the second derivative can be written as:

$$V_N''(\theta) = \frac{2}{N}\sum_{t=1}^{N} \psi_{PE}^{T}(t,\theta)\psi_{PE}(t,\theta) - \frac{2}{N}\sum_{t=1}^{N} \varepsilon(t,\theta)\frac{\partial}{\partial\theta}\psi_{PE}^{T}(t,\theta) \tag{5.39}$$

In the vicinity of the global minimum point, $\varepsilon(t,\theta)$ becomes asymptotically (as N tends to infinity) zero mean white noise which is independent of $\psi_{PE}(t,\theta)$. Then

$$V_N''(\theta_0) \approx \frac{2}{N}\sum_{t=1}^{N} \psi_{PE}^{T}(t,\theta_0)\psi_{PE}(t,\theta_0) \tag{5.40}$$

where θ_0 denotes the true parameter vector. It is appealing to neglect the second term in (5.39) when using the algorithm (5.37). There are two reasons for this:

- Generally, the approximation $V_N''(\theta)$ in (5.40) is by construction positive definite. Therefore the loss criterion function will decrease in every iteration if α_k is appropriately chosen.
- The computations are simpler since $\partial \psi_{PE}(t,\theta)/\partial\theta$ needs not be evaluated.

The algorithm obtained in this way can be written as

$$\hat{\theta}^{(k+1)} = \hat{\theta}^{(k)} + \alpha_k \left[\sum_{t=1}^{N} \psi_{PE}^{T}(t,\hat{\theta}^{(k)})\psi_{PE}(t,\hat{\theta}^{(k)})\right]^{-1}\left[\sum_{t=1}^{N} \varepsilon(t,\hat{\theta}^{(k)})\psi_{PE}^{T}(t,\hat{\theta}^{(k)})\right] \tag{5.41}$$

This is called the *Gauss-Newton algorithm*. When N is large the two algorithms (5.37) and (5.41) will behave quite similarly if $\hat{\theta}^{(k)}$ is close to the minimum point. For any N, the local convergence of the Newton-Raphson algorithm is quadratic, i.e. when $\hat{\theta}^{(k)}$ is close to $\hat{\theta}$ (the minimum point) then $\left\|\hat{\theta}^{(k+1)} - \hat{\theta}\right\|$ is of the same magnitude as $\left\|\hat{\theta}^{(k)} - \hat{\theta}\right\|^2$. The Gauss-Newton algorithm will give linear convergence in general. It will be quadratically convergent when the last term in (5.39) is zero.

To start the Newton-Raphson (5.37) or the Gauss-Newton (5.41) algorithms, an initial estimate $\hat{\theta}^{(0)}$ is required. The choice of initial value will often significantly influence the number of iterations needed to find the minimum point, and is a highly problem-dependent question. In some cases there might be *a priori* information, which then of course should be taken into account.

5.6 An Instrumental Variable (IV) Method

This section is situated here because it relies on the optimization algorithms of Section 5.5 to show how the IV can be obtained from the LS method. It was stated that the NLS method generally generates bad estimates if the disturbance is neither zero nor white. That occurs due to the correlation between $e_{LS}(t,\theta)$ and $\psi_{LS}(t,\theta)$ in (5.16). The bias due to this correlation may be avoided by modifying the NLS estimates into the Instrumental Variable (IV) method.

The idea underlying the introduction of the IV estimates of the parameter vector θ may explained as follows [1]. Assume that $Z(t)$ is an n_z column vector, the entries of which are signals uncorrelated with the disturbance $e(t)$. Then one may try to estimate the parameter vector θ of the model (5.12) and (5.13) by exploiting this property, which means that $Z(t)$ is required to satisfy the following expectations

$$\lim_{N\to\infty}\frac{1}{N}\sum_{t=1}^{N}Z(t)\psi_{LS}(t) \text{ be nonsingular} \tag{5.42}$$

$$\lim_{N\to\infty}\frac{1}{N}\sum_{t=1}^{N}e_{LS}(t)Z(t)=\mathbf{0} \tag{5.43}$$

In other words, we could say that the instruments must be correlated with $\psi_{LS}(t)$ but uncorrelated with the noise.

For linear system identification, the choice of the instrumental variable $Z(t)$, or just the instrument, is subject to certain conditions guaranteeing the unbiassedness of the estimates [1,3]. In our case, a reasonable choice of $Z(t)$ may be $\psi_{OE}^{T}(t)$, obtained in Section 5.2. Such a choice may satisfy the properties given in (5.42) and (5.43).

Apparently, there is no correlation between $\psi_{OE}^{T}(t)$ and the noisy part of $\psi_{LS}(t,\theta)$. Based on that, the IV method is a hybrid algorithm of OE and NLS algorithms and is given by

$$\hat{\theta}^{(k+1)} = \hat{\theta}^{(k)} - \alpha_k \left[\sum_{t=1}^{N} \psi_{OE}^{T}(t,\hat{\theta}^{(k)}) \psi_{LS}(t,\hat{\theta}^{(k)}) \right]^{-1} \left[\sum_{t=1}^{N} e_{LS}(t,\hat{\theta}^{(k)}) \psi_{OE}^{T}(t,\hat{\theta}^{(k)}) \right] \tag{5.44}$$

Chapter 6

IDENTIFICATION OF WIENER-HAMMERSTEIN MODELS VIA GENETIC ALGORITHMS

6.1 Introduction

Genetic Algorithms (GAs) are directed random search techniques which may find the global optimal solution in a complex multidimensional search space. GAs employ different genetic operators to manipulate individuals in a population of solutions over several generations to improve their fitness. Normally, the parameters to be optimized are represented in a binary string. However, the binary representation has some

drawbacks as the excessive computations needed when applied to multidimensional numerical optimization problems, like the case of Wiener-Hammerstein parameter identification problem. In such a case, it is better to employ the real parameter values instead when performing any coding techniques.

In the initial phase, GAs use randomly produced initial solutions, each represents a candidate parameter vector estimates, created through a random number generator. This method is preferred when *a priori* knowledge about the problem is not available.

A simple flow chart of the proposed GA is depicted in Figure 6.1. There are four genetic operators that can be used to generate and explore the neighborhood of a population and produce a new generation. These operators are reproduction, perturbation, crossover, and mutation. After randomly generating the initial population of p parameter vector estimates, the GAs use the four genetic operators to yield p new parameter vectors at each iteration, or generation. In the reproduction operation, each parameter vector of the current population is evaluated by its fitness, normally represented by the value of some objective function, and hence individuals with higher fitness values, i.e. the good parameter estimates, are selected. In the following section, the GAs operators will be discussed in details.

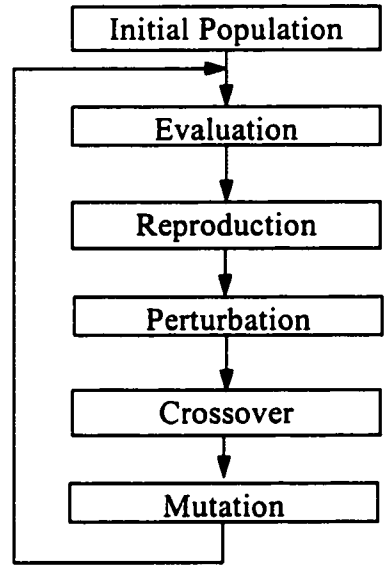


Figure 6.1. Flow chart for a simple Genetic Algorithm.

6.2 Genetic Algorithms in System Identification

Two of the attractive features of GAs are its simplicity and the multi-solution parallel optimization mechanism. Moreover, the solution is less likely to be forced into a local minimum because of the random multi-point parallel search mechanism. These two characteristics make GAs suitable for nonlinear system identification, including parameter estimation of Wiener-Hammerstein models.

Before we proceed, several terminologies used in the proposed GA should be stated.

- *Generation.* In GAs, generation, k, represents a set of candidate solutions of the optimization process. Moving from generation k to $k+1$ represents an iteration in traditional search techniques, e.g. a Gauss-Newton iteration. The number of

generations to run a GA over is pre-chosen and is generally problem dependent [42].

- *Parameters' space.* Before embarking the GAs, each parameter to be identified should have a pre-specified region in which the search of that parameter is conducted. The need of such knowledge may be considered as a drawback of the GAs, since it requires a prior information about the domain on which the true parameter value exists. It should be noted that the larger the parameter space, the more number of generations is required for the GA to converge to the optimal solution. [36,41]
- *Population and population size.* The population, $P(k)$, is defined as the collection of all the candidate strings, or parameter vectors $\hat{\theta}'s$, at the kth generation. The initial population, $P(0)$, is chosen randomly such that all the parameters fall in their respective spaces. The population size ' p ' affects both ultimate performance and efficiency of the GAs. GAs generally perform poorly with very small populations, because such populations provide insufficient sample size for most hyperplanes. A large population is more likely to contain representatives from large number of hyperplanes. Hence, the GAs can perform a more exhaustive search. As a result, a large population discourages convergence to suboptimal solutions. On the other hand, a large population requires more evaluations per generation, possibly resulting in an unacceptably slow rate of convergence [35,41]. There is a number of methods proposed in the literature based on variable population size, an example is [50], where the concept of the 'age' of an individual is introduced. The age of an individual is used to influence the size of

the population at every iteration of the algorithm. However, most of the efforts found in the literature considers a constant-size population [35,37,39,40,41,42]. A constant population size is adopted in our GAs.

- *Cost and fitness functions.* The GAs need to test each parameter vector estimate, $\hat{\theta}$, in the current population in term of a cost function in order to guide its search. In system identification, usually the following quadratic cost function is employed,

$$J(\hat{\theta}) = \sum_{t=1}^{N} \varepsilon^2(t,\hat{\theta}) \tag{6.1}$$

where $\varepsilon(t,\hat{\theta})$ is defined as the prediction error at a certain $\hat{\theta}$, t is the time index, and N is the total number of observations. The above cost function has been chosen in many of the proposed works, i.e. [39,40,42]. The GA search is guided by fitness, and therefore the value of the cost function should be mapped into a fitness function. The fitness value f is a measure of the cost function. It can be linear or nonlinear, differentiable or non-differentiable, continuous or discontinuous, but a positive function [37]. The smaller the cost function value is, the higher the fitness will be. Thus, the fitness function should be inversely proportional to the cost function. A popular fitness function is

$$f(\hat{\theta}) = \frac{1}{J(\hat{\theta})+1} \tag{6.2}$$

The above fitness function has been used for its simplicity. It implies that if the cost is very small, the fitness approaches the value of 1. If the cost is very large, the fitness approaches the value of zero.

- *Reproduction.* Reproduction is based on the principle of survival of the fittest. A fitness, $f(\hat{\theta})$, is assigned to each estimated parameter vector in the population. Based on a normalized fitness $f_n(\hat{\theta})$, a number of copies for each $\hat{\theta}$ is calculated. In other words, the fitness is normalized with the sum of the fitness values, $f_n(\hat{\theta}) = f(\hat{\theta}) / \sum_{i=1}^{p} f(\hat{\theta}_i)$. The cumulative sum of these normalized values is then calculated and stored in $F(\hat{\theta})$. Based on the Roulette Wheel Selection (RWS), an estimated parameter vector $\hat{\theta}$ generates copies with probability equal to the value of $F(\hat{\theta})$, which ranges from 0 to 1. [37,42].
- *Perturbation.* This is a novel operator in system identification via GAs. Perturbation is a random addition, with a small probability, to the value of the parameter vector $\hat{\theta}$ according to the value of $f(\hat{\theta})$. This addition is taken from a normal distribution with a zero mean and a variance equal to $1 - f(\hat{\theta})$. This operator takes place in order to enrich the diversity of the population and hence preventing the GA from being trapped in a local minimum. The perturbation probability, P_p, is normally a small value and it is chosen such that at least one $\hat{\theta}$ string in the population is perturbed.
- *Crossover.* Reproduction directs the search towards the best existing parameter vectors, or strings, but does not create new strings. In nature, an offspring is rarely an exact clone of a parent, it usually has two parents and inherits genes from both. The main operator to work on the parents is crossover, which is applied with certain probability, called the crossover probability P_c. P_c usually has a high

value in order to allow more frequent information exchange between fit $\hat{\theta}$'s. This operator takes valuable information from both parents and combines it to find a better candidate. To apply this operator, two different parameter vectors from the reproduced population are mated at random and are cut randomly between any two existing $\hat{\theta}$'s. The new $\hat{\theta}$'s are created by interchanging their tails. It should be mentioned here that reproduction and crossover give GAs much of their power by directing the search toward the better areas using already existing knowledge. [37,39,41]

- *Mutation.* Mutation is a secondary search operator which increases the variability of the population. According to the value of $J(\hat{\theta})$, mutation is used to slightly alter, or change, $\hat{\theta}$ or a single parameter within $\hat{\theta}$. When using the real values of the parameters, mutation is applied in two different ways, according to the value of $J(\hat{\theta})$. If $J(\hat{\theta})$ is low, then a little mutation is conducted by interchanging two digits in the parameter value. That is, if the parameter has the value of 0.4567, then after mutation it may have the value 0.5467. Otherwise, if $J(\hat{\theta})$ is high, mutation is applied by flipping the parameter's sign. The mutation operator is applied with a certain probability, called mutation probability, P_m. Such probability should be kept small. [35,37,41]
- *Filling.* The perturbation, crossover, and mutation operators are conducted on the best parameter vectors that contribute an s % of p; where $s = 25$. After conducting these operators, $0.75 \times p$ new parameter vectors are randomly

generated and inserted in $P(k+1)$ in order to keep a constant population size. The reason of adding new $\hat{\theta}'s$ in P is to enrich the diversity of the adopted GAs.

Using parameter real values without any encoding in the reproduction phase, employing the perturbation operator, and relating mutation to the fitness of the parameter being altered make the adopted GA a new one. Appendix A gives the pseudocodes of the GAs mentioned above. One last point to mention about the adopted Genetic Algorithm is that the best parameter estimates vector is passed unchanged into the next generation in order to assure that the minimum estimation error is a monotonically decreasing process.

The following flow chart, (Figure 6.2), shows the steps to be followed in order to apply our GA to a Wiener-Hammerstein model parameter identification. The variables used in the flow chart are defined in Appendix A.

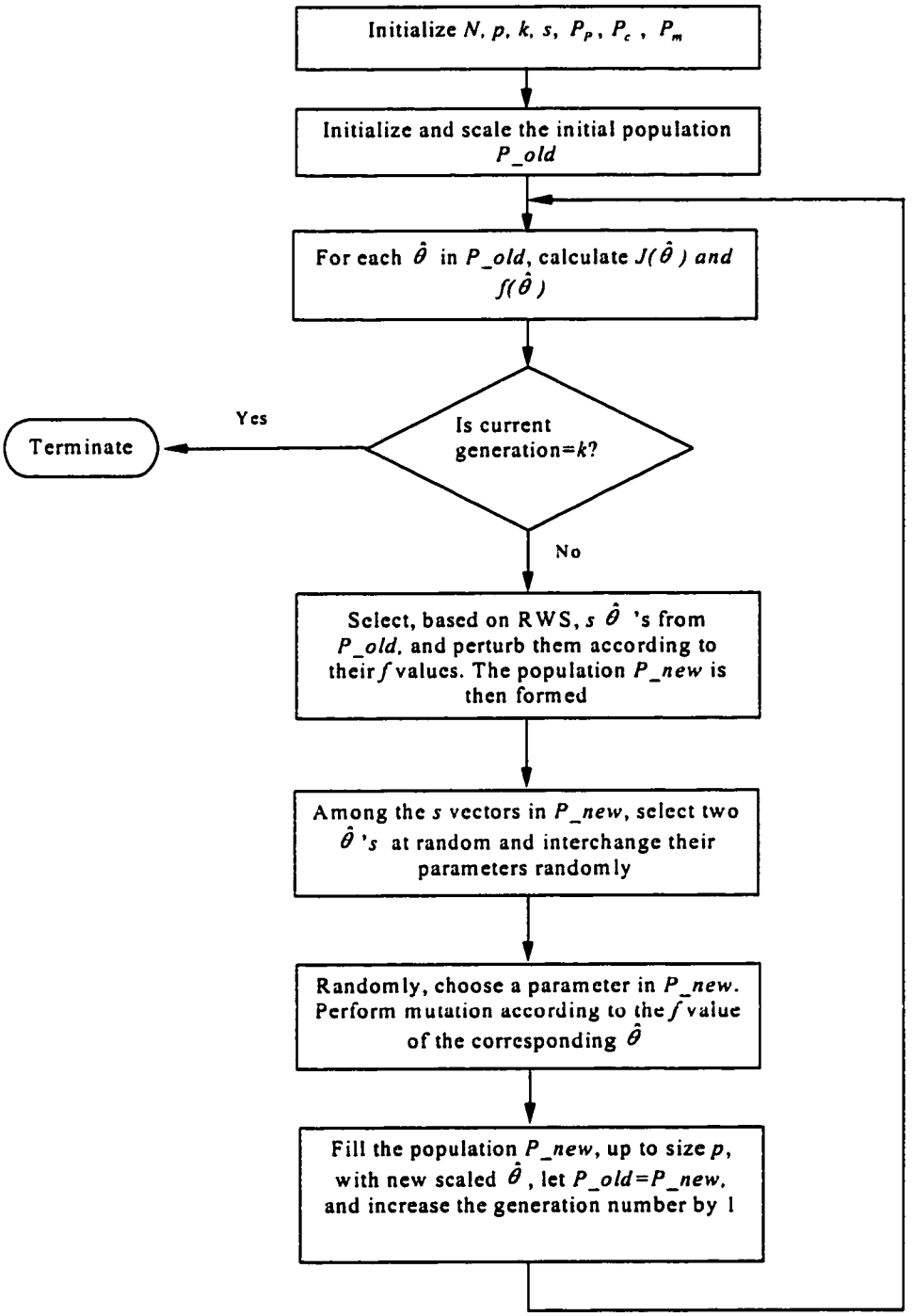


Figure 6.2. Detailed flow chart for the adopted Genetic Algorithm

6.3 A Search Example for a Two-Parameter Estimation Problem [38]

Consider a nonlinear system with the following characteristics:

$$x_1(t+1) = \sin(a_1 x_1(t)) + \sin(a_2 x_2(t)) + 0.01\sin(a_1 x_1(t)u(t))$$
$$x_2(t+1) = \sin((a_1 + a_2)^2 x_2(t)) + 0.01 a_1^2 x_2(t)u(t)$$
$$y(t) = x_1(t) + x_2(t)$$

The problem under consideration is to estimate the parameters a_1 and a_2 for this nonlinear system. For our test case, we arbitrarily set the 'true' values for these parameters to be $a_1 = -1.0, a_2 = 1.0$ and generate a discrete sequence of input data using the input signal $u(t) = \sin(t) + \sin(t/3) + \sin(t/5)$ and a corresponding output sequence $y(t)$.

Using these data sets as the reference models, we define a two-dimensional parameter space $-2.0 \le a_i \le 2.0$, $i = 1,2$, to be searched by the GAs in order to find the 'best' estimate of the parameters. The fitness of a particular estimate is obtained by observing the behavior of the system with the estimated parameters, and employing the Sum Square Error (SSE) between the model output and the observed data.

To get a sense of the difficulty of the estimation problem, we plotted the error surface just for a small number of points in the search space, as illustrated in Figure 6.3.

We then use the adopted GAs with a population of size 50 (along with a perturbation probability of 0.01, crossover probability of 0.6, and a mutation

probability of 0.1) to search in the specified parameter space. These values have been set via simulation.

Figures 6.4, 6.5, and 6.6 represent the population after certain number of generations. The 'x's represents the population points in the parameter spaces, and the '*' indicates the position of the optimum point. While Figures 6.7 and 6.8 represent the best parameter estimates and SSE values per generation.

As can be observed from Figure 6.4, at the initial generation, the population is randomly distributed in the whole parameter space. During the evolution of the algorithms, as depicted in Figure 6.5, the elements of the population tend to concentrate at the minima of the function and at convergence, Figure 6.6, most of the estimates are clustered near the desired minimum value of the function.

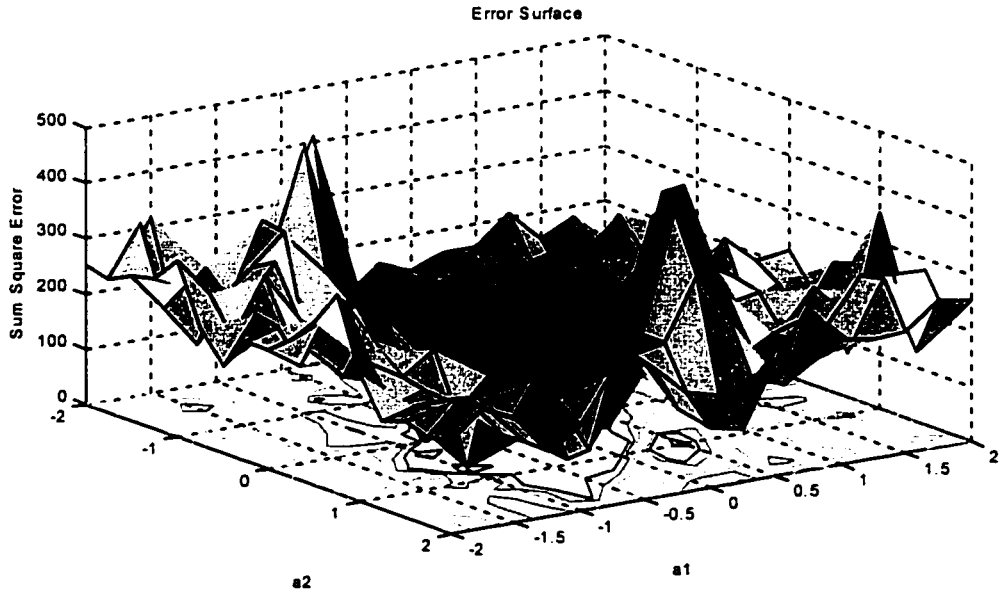


Figure 6.3. Multimodality of the SSE surface

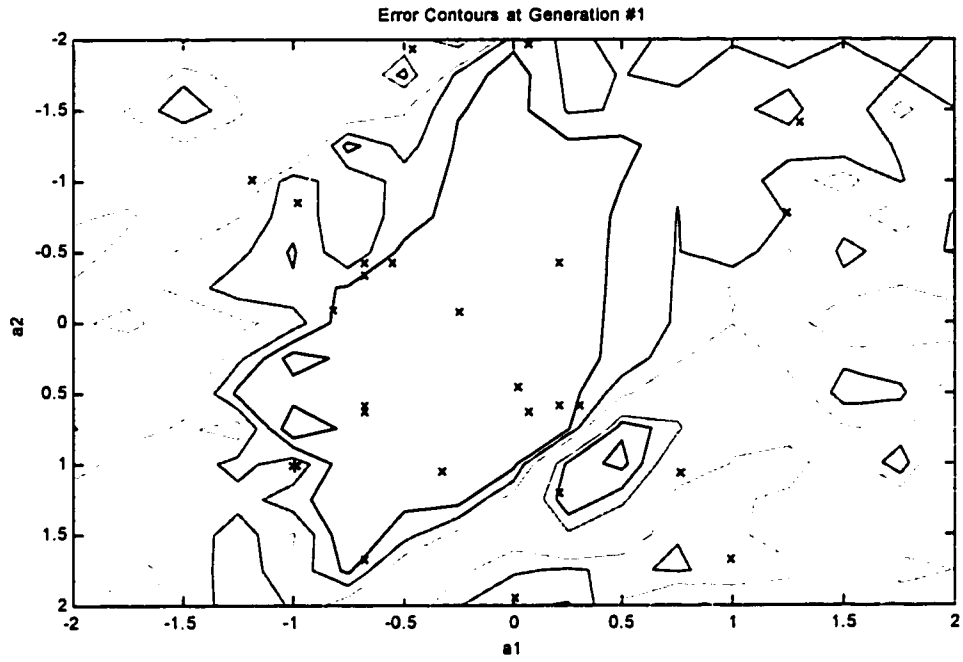


Figure 6.4. The initial population

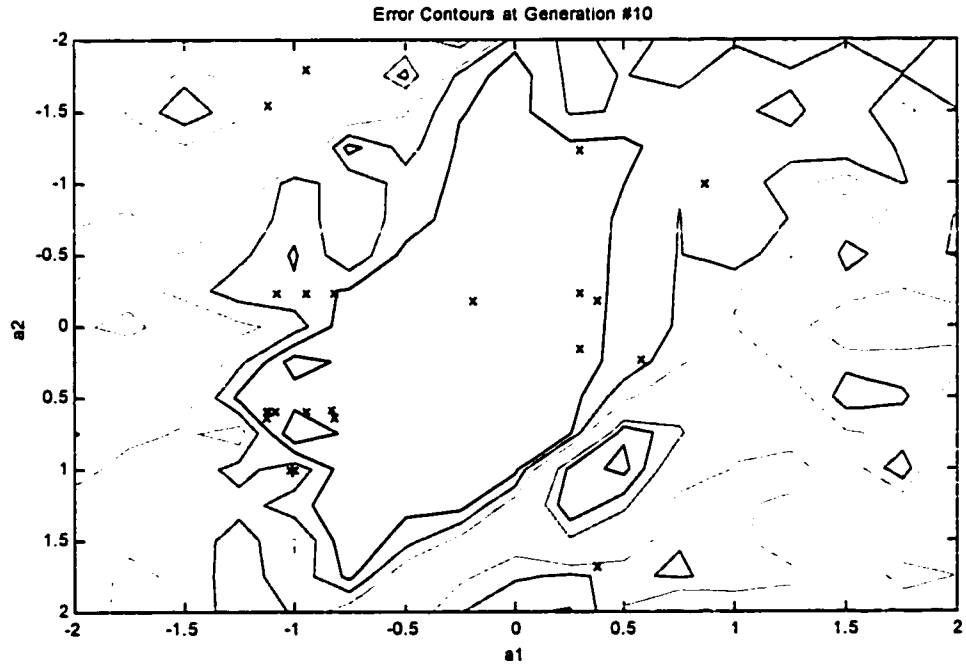


Figure 6.5. Population at generation # 10

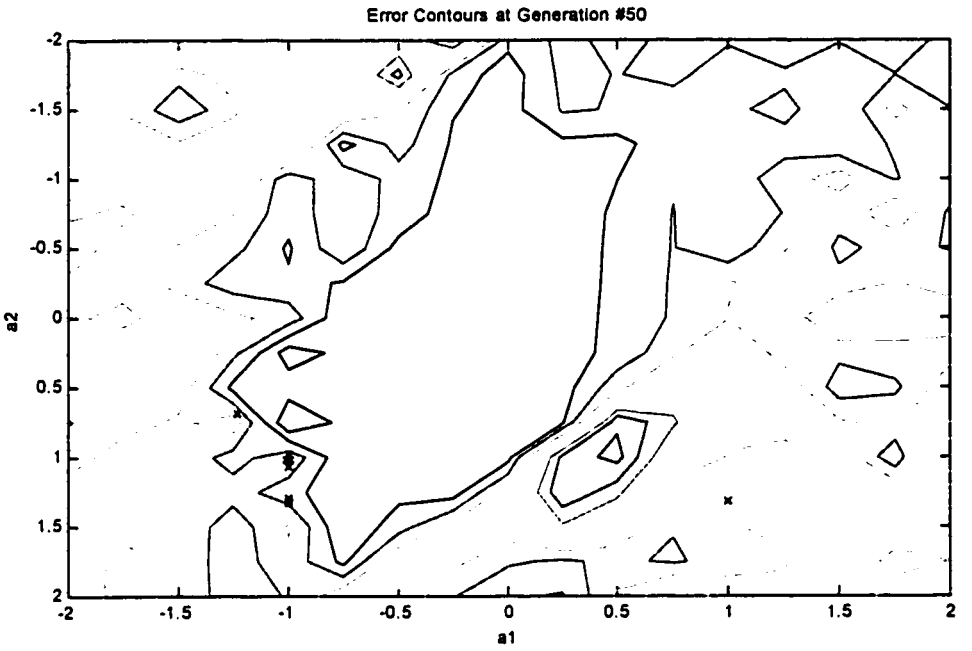


Figure 6.6. Population at generation # 50

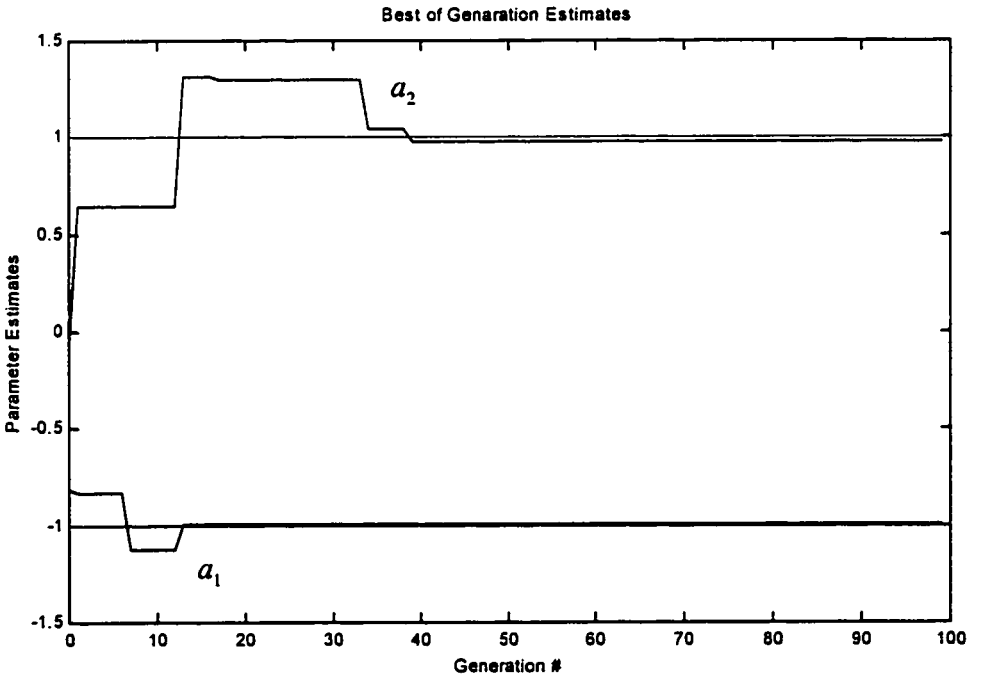


Figure 6.7. The best parameter estimates per generation

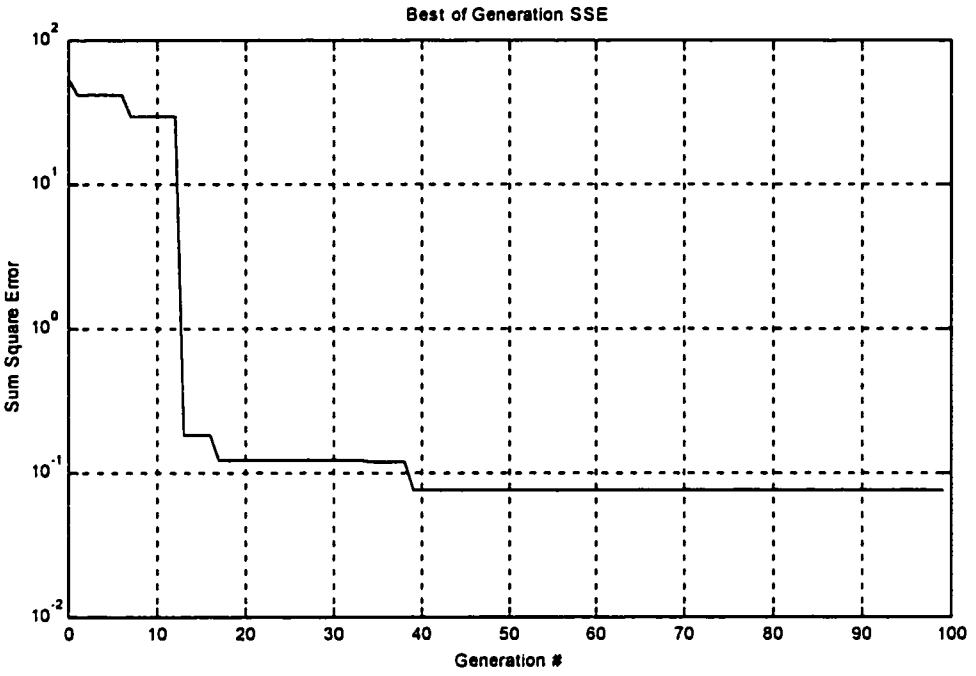


Figure 6.8. The best SSE per generation

Chapter 7

ONLINE IDENTIFICATION OF WIENER-HAMMERSTEIN MODELS

7.1 Introduction

In online (also called recursive) identification methods, the parameter estimates are updated as new data arrive. This means that if an estimate $\hat{\theta}(t-1)$ based on data up to time $t-1$ is available, then $\hat{\theta}(t)$ is computed employing $\hat{\theta}(t-1)$ and the data that arrive at time t. The counterparts of the online methods are the so-called offline or

batch methods, in which all the recorded data are used simultaneously to find the parameter estimates. The offline methods were discussed in Chapter 5.

Generally, recursive identification methods have the following features:

- They are the central part of adaptive systems (used, for example, for control or signal processing) where the action (control, filtering, etc.) is based on the most recent model.
- Their requirement on primary memory is quite modest, since not all data are stored.
- They can be easily modified into real-time algorithms, aimed at tracking time-varying parameters.
- They can be the first step in a fault detection algorithm, which is used to find out whether the system has changed significantly. [1]

Many recursive identification methods are derived as approximations of their offline methods. It may therefore happen that the price paid for the approximation is a reduction in accuracy. It should be noted, however, that the user seldom chooses between offline and online methods, but rather between different offline methods or different online methods. In this chapter, the offline methods of Chapter 5 are used to derive their recursive counterparts. Consequently, we will have the following four recursive identification techniques, namely Recursive Prediction Error (RPE), Recursive Output Error (ROE), Recursive Nonlinear Least Squares (RNLS), and the Recursive Instrumental Variable (RIV). In addition, a computationally simple Stochastic Approximation (SA) method is also considered. [1,8]

7.2 Recursive Prediction Error (RPE) Algorithm [1]

The derivations of all the recursive methods in this chapter are quite similar to each other. Therefore, we will consider only the derivation of the RPE, which is established in the sequel.

The formulation of the Recursive Prediction Error algorithm includes the use of a forgetting factor λ. The following criterion function is considered

$$V_t(\theta) = \frac{1}{2}\sum_{k=1}^{t} \lambda^{t-k}\varepsilon^2(k,\theta) \tag{7.1}$$

where $\varepsilon(t)$ is defined in equation (5.27). The estimates $\hat{\theta}_t$ which minimizes $V_t(\theta)$ cannot be found analytically. Instead, a numerical optimization must be performed. Derivation of such recursive optimization schemes requires further approximations.

Assume that $\hat{\theta}(t-1)$ minimizes $V_{t-1}(\theta)$ and that the minimum of $V_t(\theta)$ is close to $\hat{\theta}(t-1)$. Then, it is reasonable to approximate $V_t(\theta)$ by a second-order Taylor's series expansion around $\hat{\theta}(t-1)$:

$$\begin{aligned} V_t(\theta) \approx & V_t(\hat{\theta}(t-1)) + [V_t'(\hat{\theta}(t-1))]^{\mathrm{T}}(\theta - \hat{\theta}(t-1)) \\ & + \frac{1}{2}(\theta - \hat{\theta}(t-1))^{\mathrm{T}} V_t''(\hat{\theta}(t-1))(\theta - \hat{\theta}(t-1)) \end{aligned} \tag{7.2}$$

The right-hand side of (7.2) is a quadratic function of θ. Minimize this with respect to θ and let the minimum point constitute the new parameter estimate $\hat{\theta}(t)$. Thus

$$\hat{\theta}(t) = \hat{\theta}(t-1) - [V_t''(\hat{\theta}(t-1))]^{-1}[V_t'(\hat{\theta}(t-1))] \tag{7.3}$$

which corresponds to one step of the Newton-Raphson algorithm (5.37) initialized at $\hat{\theta}(t-1)$. In order to proceed further, recursive relationships for the loss function $V_t(\theta)$ and its derivative are needed. From (7.1) , one may obtain

$$V_t(\theta) = \lambda V_{t-1}(\theta) + \frac{1}{2}\varepsilon^2(t,\theta) \tag{7.4}$$

$$V_t'(\theta) = \lambda V_{t-1}'(\theta) + \varepsilon(t,\theta)\varepsilon'(t,\theta) \tag{7.5}$$

$$V_t''(\theta) = \lambda V_{t-1}''(\theta) + \varepsilon'(t,\theta)[\varepsilon'(t,\theta)]^{\mathrm{T}} + \varepsilon(t,\theta)\varepsilon''(t,\theta) \tag{7.6}$$

where $\varepsilon(t,\theta)$ is scalar, $\varepsilon'(t,\theta)$ is a column Gradient vector, and $\varepsilon''(t,\theta)$ is the Hessian matrix.

Now, make the following approximations:

$$V_{t-1}'(\hat{\theta}(t-1)) = \mathbf{0} \tag{7.7}$$

$$V_{t-1}''(\hat{\theta}(t-1)) = V_{t-1}''(\hat{\theta}(t-2)) \tag{7.8}$$

$\varepsilon(t,\theta)\varepsilon''(t,\theta)$ in (7.6) is negligible (7.9)

The motivation for (7.7) is that $\hat{\theta}(t-1)$ is assumed the minimum point of $V_{t-1}(\theta)$. The approximation (7.8) means that the second-order derivative $V_{t-1}''(\theta)$ varies slowly with θ. The reason for (7.9) is that $\varepsilon(t,\theta)$ at the true parameter vector θ_0 will be a white sequence and hence

$$E\{\varepsilon(t,\theta)\varepsilon''(t,\theta)\} = \mathbf{0} \tag{7.10}$$

This implies that at least for large t and θ close to the minimum point, one can indeed neglect the influence of the last term in (7.6) on $V_t''(\theta)$.

Using (7.4)-(7.9), the following algorithm is derived from (7.3)

$$\hat{\theta}(t) = \hat{\theta}(t-1) - \varepsilon(t,\hat{\theta}(t-1))[V_t''(\hat{\theta}(t-1))]^{-1}[\varepsilon'(t,\hat{\theta}(t-1))] \tag{7.11}$$

$$V_t''(\hat{\theta}(t-1)) = \lambda V_{t-1}''(\hat{\theta}(t-2)) + \varepsilon'(t,\hat{\theta}(t-1))[\varepsilon'(t,\hat{\theta}(t-1))]^{\mathrm{T}} \tag{7.12}$$

This algorithm as it stands is not well suited as a recursive algorithm for the following reason:

- The inverse of V_t'' is needed in (7.11), while the matrix itself (and not its inverse), is updated in (7.12).

This problem can be solved by applying the matrix inversion lemma to (7.12), as described below. In order to simplify the derivation, we use the following notation, let

$$\varepsilon(t) \approx \varepsilon(t,\hat{\theta}(t-1)) \tag{7.13}$$

$$\psi_{PE}(t) \approx -[\varepsilon'(t,\hat{\theta}(t-1))]^{\mathrm{T}} \tag{7.14}$$

that can be evaluated online.

Now, introduce the covariance matrix $P_{PE}(t)$ as

$$P_{PE}(t) = [V_{t-1}''(\hat{\theta}(t-1))]^{-1} \tag{7.15}$$

Then from (7.12),

$$P_{PE}^{-1}(t) = \lambda P_{PE}^{-1}(t-1) + \psi_{PE}^{\mathrm{T}}(t)\psi_{PE}(t) \tag{7.16}$$

which can be rewritten by using the matrix inversion lemma (Appendix B), to give

$$P_{PE}(t) = \frac{1}{\lambda}\left\{P_{PE}(t-1) - \frac{P_{PE}(t-1)\psi_{PE}^{\mathrm{T}}(t)\psi_{PE}(t)P_{PE}(t-1)}{\psi_{PE}(t)P_{PE}(t-1)\psi_{PE}^{\mathrm{T}}(t) + \lambda}\right\} \tag{7.17}$$

Then, the Recursive Prediction Error (RPE) algorithm for a Wiener-Hammerstein model becomes

$$\hat{\theta}(t) = \hat{\theta}(t-1) + K(t)\varepsilon(t)$$
$$K(t) = P_{PE}(t)\psi_{PE}^{\mathrm{T}}(t) \tag{7.18}$$
$$P_{PE}(t) = \frac{1}{\lambda}\left\{P_{PE}(t-1) - \frac{P_{PE}(t-1)\psi_{PE}^{\mathrm{T}}(t)\psi_{PE}(t)P_{PE}(t-1)}{\psi_{PE}(t)P_{PE}(t-1)\psi_{PE}^{\mathrm{T}}(t) + \lambda}\right\}$$

The gain $K(t)$ can be rewritten in a more convenient computational form. From (7.17), we obtain

$$\begin{aligned} K(t) &= P_{PE}(t-1)\psi_{PE}^{\mathrm{T}}(t)/\lambda - P_{PE}(t-1)\psi_{PE}^{\mathrm{T}}(t)[\lambda + \psi_{PE}(t)P_{PE}(t-1)\psi_{PE}^{\mathrm{T}}(t)]^{-1} \\ &\quad \times \psi_{PE}(t)P_{PE}(t-1)\psi_{PE}^{\mathrm{T}}(t)/\lambda \\ &= P_{PE}(t-1)\psi_{PE}^{\mathrm{T}}(t)/\lambda[\lambda + \psi_{PE}(t)P_{PE}(t-1)\psi_{PE}^{\mathrm{T}}(t)]^{-1} \\ &\quad \times \{[\lambda + \psi_{PE}(t)P_{PE}(t-1)\psi_{E}^{\mathrm{T}}(t)] - \psi_{PE}(t)P_{PE}(t-1)\psi_{PE}^{\mathrm{T}}(t)\} \\ &= P_{PE}(t-1)\psi_{PE}^{\mathrm{T}}(t)[\lambda + \psi_{PE}(t)P_{PE}(t-1)\psi_{PE}^{\mathrm{T}}(t)]^{-1} \end{aligned}$$

that is,

$$K(t) = P_{PE}(t-1)\psi_{PE}^{\mathrm{T}}(t)[\lambda + \psi_{PE}(t)P_{PE}(t-1)\psi_{PE}^{\mathrm{T}}(t)]^{-1} \tag{7.19}$$

7.3 Recursive Output Error (ROE) Algorithm

Following the discussions and derivations in the previous section, the Recursive Output Error (ROE) algorithm used to estimate the parameters of the model under study can be obtained as:

$$\hat{\theta}(t) = \hat{\theta}(t-1) + K(t)e_{OE}(t)$$
$$K(t) = P_{OE}(t)\psi_{OE}^{\mathrm{T}}(t) \tag{7.20}$$
$$P_{OE}(t) = \frac{1}{\lambda}\left\{P_{OE}(t-1) - \frac{P_{OE}(t-1)\psi_{OE}^{\mathrm{T}}(t)\psi_{OE}(t)P_{OE}(t-1)}{\psi_{OE}(t)P_{OE}(t-1)\psi_{OE}^{\mathrm{T}}(t) + \lambda}\right\}$$

where $e_{OE}(t)$ is as defined earlier in equation (5.3), in the offline scheme,

$\psi_{OE}(t) \approx -[e'_{OE}(t, \hat{\theta}(t-1))]^{\mathsf{T}}$,

and

$P_{OE}^{-1}(t) = \lambda P_{OE}^{-1}(t-1) + \psi_{OE}^{\mathsf{T}}(t)\psi_{OE}(t)$

7.4 Recursive Nonlinear Least Squares (RNLS) Algorithm

Following the derivations in Section 7.2, the Recursive Nonlinear Least Squares (RNLS) algorithm can be obtained as follows:

$$\begin{aligned} \hat{\theta}(t) &= \hat{\theta}(t-1) + K(t)e_{LS}(t) \\ K(t) &= P_{LS}(t)\psi_{LS}^{\mathsf{T}}(t) \\ P_{LS}(t) &= \frac{1}{\lambda}\left\{P_{LS}(t-1) - \frac{P_{LS}(t-1)\psi_{LS}^{\mathsf{T}}(t)\psi_{LS}(t)P_{LS}(t-1)}{\psi_{LS}(t)P_{LS}(t-1)\psi_{LS}^{\mathsf{T}}(t) + \lambda}\right\} \end{aligned} \tag{7.21}$$

where

$\psi_{LS}(t) \approx -[e'_{LS}(t, \hat{\theta}(t-1))]^{\mathsf{T}}$,

and

$P_{LS}^{-1}(t) = \lambda P_{LS}^{-1}(t-1) + \psi_{LS}^{\mathsf{T}}(t)\psi_{LS}(t)$

7.5 Recursive Instrumental Variable (RIV) Algorithm

Following the approach of earlier sections, the parameter estimate of the adopted Wiener-Hammerstein model can be obtained via the Recursive Instrumental Variable (RIV) algorithm as:

$$\begin{aligned}
&\hat{\theta}(t) = \hat{\theta}(t-1) + K(t)e_{LS}(t) \\
&K(t) = P_{IV}(t)Z(t) \\
&P_{IV}(t) = \frac{1}{\lambda}\left\{P_{IV}(t-1) - \frac{P_{IV}(t-1)Z(t)\psi_{LS}(t)P_{IV}(t-1)}{\psi_{LS}(t)P_{IV}(t-1)Z(t) + \lambda}\right\}
\end{aligned} \tag{7.22}$$

where $Z(t)$ is an instrumental vector. As has been selected in the offline IV method, $Z(t)$ may be taken as $\psi_{OE}^{\mathrm{T}}(t)$ where ψ_{OE} is defined in equation (5.5).

7.6 Stochastic Approximation (SA) Algorithm

The heaviest computational task in the above recursive algorithms is updating the covariance matrices (i.e. $P_{PE}(t), P_{OE}(t), P_{LS}(t), P_{IV}(t)$). Studies have been conducted to cut down on computation by bypassing this recursion. In this regard, attempts have been made to calculate the correction gain $K(t)$ without updating the covariance matrix. However, the price may be a larger variance of the parameter estimates. [3]

The simplest scheme would be to correct the prediction error $\varepsilon(t)$ by a pre-determined positive scalar $\gamma(t)$ multiplied by $\psi_{PE}(t)$, giving a so-called Stochastic Approximation (SA) algorithm: i.e.

$$\hat{\theta}(t) = \hat{\theta}(t-1) + \gamma(t)\psi_{PE}^{\mathrm{T}}(t)\varepsilon(t) \tag{7.23}$$

The step-size factors γ have to be both large enough for $\hat{\theta}$ to reach the correct value and small enough finally for $\hat{\theta}$ to stay there. There is a large literature on how to select the gain sequence $\gamma(t)$ [3]. The theory of stochastic approximation usually leaves the choice of the gain sequence $\gamma(t)$ open, as long as it satisfies

$$\gamma(t) \geq 0, \sum_{t=1}^{\infty}\gamma(t) = \infty, \sum_{t=1}^{\infty}\gamma^2(t) < \infty.$$

7.7 Initial Values

To use the recursive algorithms, initial values for $\hat{\theta}$ and P are needed. It will be shown later that $P(t)$ is related to covariance of $\hat{\theta}(t)$. Thus it is reasonable to take for $\hat{\theta}(0)$ an *a priori* estimate of θ and let $P(0)$ reflect the confidence in this initial estimate $\hat{\theta}(0)$. For the linear system identification, it is known that if $P(0)$ is chosen small then $K(t)$ as well becomes small for all t and the parameter estimates do not change too much from $\hat{\theta}(0)$. On the other hand, if $P(0)$ is large, the parameter

estimates quickly move away from $\hat{\theta}(0)$. Without any *a priori* information it is common practice to take

$$\hat{\theta}(0) = \mathbf{0} \qquad P(0) = \rho \mathbf{I} \tag{7.24}$$

with $\rho >> 0$.

However, for the case of Wiener-Hammerstein model identification the algorithm is highly nonlinear and a large value of ρ may drive the algorithm unstable. A good choice of ρ has been found through simulation as

$$10^{-1} > \rho > 0 \tag{7.25}$$

7.8 Forgetting Factor

The choice of the forgetting factor λ in the recursive algorithms is often very important. Theoretically, for linear system identification, one must have $\lambda = 1$ to get convergence [1]. On the other hand, with $\lambda < 1$, as t increases the measurements obtained previously are discounted. The smaller the value of λ is, the quicker the information in previous data are forgotten. However, the nonlinear identification algorithms derived in this chapter rely on the assumption that $\hat{\theta}(0)$ is close to θ_0, which generally is not true. As a result more weight are needed to be given on data used in the later stage when $\hat{\theta}(t)$ approaches the vicinity of θ_0. It has been validated through simulation that a suitable value of λ in Wiener-Hammerstein parameter recursive estimation is 0.99.

Chapter 8

CONVERGENCE ANALYSIS

8.1 Introduction

In Chapters 2, 5, and 7 we have discussed a framework for developing and describing offline and recursive identification algorithms for the Wiener-Hammerstein models. We have also studied a number of specific schemes. We now turn to the logical question: How do these algorithms perform, i.e., what are the properties of the estimates $\hat{\theta}$ or $\hat{\theta}(t)$?

Generally, such a question can be answered in two ways. The first way is to apply the algorithms to known data sequences and examine the obtained estimates. This is know as *simulation*. Simulation is a very useful tool for investigating identification algorithms. In Chapter 9, all the identification algorithms will be tested through simulation on two examples. However, a limitation of simulation is that it may not be conclusive. It is sometimes hard to tell whether a simulation result has universal implications, or merely reflects properties of the chosen data sequence. To obtain results of more general validity one must use *analysis*. In this chapter, we make certain assumptions about the data, i.e. input/output pairs, as well as the Wiener-Hammerstein model structure and try to project the resulting properties of the estimates $\hat{\theta}$ or $\hat{\theta}(t)$.

It is important to mention herein that very few trials have been established in the literature to study and analyze the properties of the estimates based of the nonlinear models of Wiener-Hammerstein. Boutayeb *et al* in [32] have proposed convergence analysis of the RLS identification of Wiener-Hammerstein model.

8.2 Basic Assumptions

In order to proceed with our analysis, we make the following assumptions

1. The linear systems $\frac{B(q^{-1})}{A(q^{-1})}$, $\frac{D(q^{-1})}{C(q^{-1})}$, and $\frac{F(q^{-1})}{A(q^{-1})}$ are considered to be stable and differentiable functions of the parameter θ.

2. No pole-zero cancellation in $A(q^{-1})$ and $B(q^{-1})$, $C(q^{-1})$ and $D(q^{-1})$, and $A(q^{-1})$ and $F(q^{-1})$.
3. The polynomials $B(q^{-1})$ and $D(q^{-1})$ are monic.
4. The static nonlinearity polynomial is non-monic and single-valued in the region where the input/output data span.
5. The data $\{u(t), y(t)\}$ are stationary processes.
6. The input $u(t)$ is persistently exciting of sufficiently high order.
7. The Hessian $V_N''(\theta)$ is nonsingular at least locally around the minimum points of $V_N(\theta)$.
8. The criterion function $V_N(\theta)$ in unimodal; i.e. it has only one optimum point, $V_N(\theta_0)$, where θ_0 is the true parameter vector.

Although the first assumption will further restrict the domain of Wiener-Hammerstein models, it is imposed in order to make sure that the poles of the linear systems are stable resulting in stable identification algorithms. Assumptions 3 and 4 state that, without loss of generality, the DC gain of the system can be absorbed by the static nonlinearity polynomial.

8.3 Consistency Analysis

The following analysis is based on the PE algorithm. Refer to equation (5.24). i.e.

$$A(q^{-1})y(t) = B(q^{-1})w(t) + F(q^{-1})\varepsilon(t,\theta) \tag{8.1}$$

Rearrange the above equation to get

$$\varepsilon(t,\theta) = F^{-1}(q^{-1})A(q^{-1})y(t) - F^{-1}(q^{-1})B(q^{-1})w(t) \tag{8.2}$$

where $\varepsilon(t,\theta)$ is the prediction error sequence and θ is any arbitrary parameter vector.

Now, let θ_0 be the true parameter vector. This means that the true system satisfies

$$\begin{aligned} y(t) &= \frac{B_0(q^{-1})}{A_0(q^{-1})}w(t) + \frac{F_0(q^{-1})}{A_0(q^{-1})}\varepsilon(t,\theta_0) \\ &= \frac{B_0(q^{-1})}{A_0(q^{-1})}\left[\sum_{l=1}^{m} g_{0l}\left(\frac{D_0(q^{-1})}{C_0(q^{-1})}u(t)\right)^l\right] + \frac{F_0(q^{-1})}{A_0(q^{-1})}\varepsilon(t,\theta_0) \end{aligned} \tag{8.3}$$

Now, equation (8.3) becomes

$$\begin{aligned} y(t) &= \frac{B_0(q^{-1})}{A_0(q^{-1})}w(t) + \frac{F_0(q^{-1})}{A_0(q^{-1})}\varepsilon(t,\theta_0) \\ &= \frac{B_0(q^{-1})}{A_0(q^{-1})}\left[\sum_{l=1}^{m} g_{0l}\left(\frac{D_0(q^{-1})}{C_0(q^{-1})}u(t)\right)^l\right] + \frac{F_0(q^{-1})}{A_0(q^{-1})}\varepsilon(t,\theta_0) \end{aligned} \tag{8.4}$$

By substituting (8.4) in (8.2), we obtain

$$\begin{aligned} \varepsilon(t,\theta) &= F^{-1}(q^{-1})A(q^{-1})\left\{\frac{B_0(q^{-1})}{A_0(q^{-1})}\sum_{l=1}^{m} g_{0l}\left(\frac{D_0(q^{-1})}{C_0(q^{-1})}u(t)\right)^l + \frac{F_0(q^{-1})}{A_0(q^{-1})}\varepsilon(t,\theta_0)\right\} \\ &\quad -F^{-1}(q^{-1})B(q^{-1})\left\{\sum_{l=1}^{m} g_{0l}\left(\frac{D_0(q^{-1})}{C_0(q^{-1})}u(t)\right)^l\right\} \\ &= F^{-1}(q^{-1})\left\{A(q^{-1})\frac{B_0(q^{-1})}{A_0(q^{-1})}\sum_{l=1}^{m} g_{0l}\left(\frac{D_0(q^{-1})}{C_0(q^{-1})}u(t)\right)^l - B(q^{-1})\sum_{l=1}^{m} g_l\left(\frac{D(q^{-1})}{C(q^{-1})}u(t)\right)^l\right\} \\ &\quad +F^{-1}(q^{-1})A(q^{-1})\frac{F_0(q^{-1})}{A_0(q^{-1})}\varepsilon(t,\theta_0) \end{aligned} \tag{8.5}$$

Notice that $u(t)$ and $e(t)$ are independent. Hence if $A = A_0$, $B = B_0$, $C = C_0$, $D = D_0$, $F = F_0$, and $g_l = g_{0l}$ in (8.5), then by employing assumption 8,

$\varepsilon(t,\theta) = \varepsilon(t,\theta_0)$. □

8.4 Convergence Analysis Using Newton-Raphson Method

The following analysis is made in order to establish quadratic convergence of PE of Section 5.4 by means of the Newton-Raphson method. Investigating convergence of OE, NLS, and IV can be made in similar ways.

To start with, define

$$\begin{aligned} \gamma^{(k)} &= \hat{\theta}^{(k)} - \theta_0 \\ \gamma^{(k+1)} &= \hat{\theta}^{(k+1)} - \theta_0 \end{aligned} \tag{8.6}$$

where θ_0 is the true parameter vector and $\hat{\theta}^{(k)}$ is the parameter estimate vector at the kth Newton-Raphson algorithm. $\hat{\theta}^{(k)}$ is defined as

$$\hat{\theta}^{(k)} = (a_1^{(k)} \ldots a_{n_a}^{(k)}\, b_1^{(k)} \ldots b_{n_b}^{(k)}\, c_1^{(k)} \ldots c_{n_c}^{(k)}\, d_1^{(k)} \ldots d_{n_d}^{(k)}\, f_1^{(k)} \ldots f_{n_f}^{(k)}\, g_1^{(k)} \ldots g_m^{(k)})^{\mathrm{T}}$$

In order to proceed easily, the vector $\theta^{(k)}$ may be redefined as follows

$$\hat{\theta}^{(k)} = (\hat{\theta}_1^{(k)}\, \hat{\theta}_2^{(k)} \ldots \hat{\theta}_{n_R}^{(k)})^{\mathrm{T}} \tag{8.7}$$

where $n_R = n_a + n_b + n_c + n_d + n_f + m$.

Then, using Newton-Raphson algorithm, i.e. equation (5.37), while setting α_k to 1, yields

$$\begin{aligned} \gamma^{(k+1)} &= \hat{\theta}^{(k)} - [V_N''(\hat{\theta}^{(k)})]^{-1} V_N'(\hat{\theta}^{(k)}) - \theta_0 \\ &= \gamma^{(k)} - [V_N''(\hat{\theta}^{(k)})]^{-1} V_N'(\hat{\theta}^{(k)}) \end{aligned} \tag{8.8}$$

By employing assumption 8, where θ_0 is an optimal vector of $V_N(\theta)$, we obtain

$$V_N'(\theta_0) = \mathbf{0} \tag{8.9}$$

Then by Multiple Taylor's Theorem [51], there exists an $n_R \times 1$ vector $\tilde{\theta}^{(k)}$ such that

$$\tilde{\theta}^{(k)}(i) \in [\hat{\theta}^{(k)}(i), \theta_0(i)] \quad , \text{ for } i = 1,2,\cdots,n_R \tag{8.10}$$

And the Taylor series expansion associated with $V_N'(\theta_0)$ becomes [51]

$$\mathbf{0} = V_N'(\theta_0) = V_N'(\hat{\theta}^{(k)} - \gamma^{(k)}) \approx V_N'(\hat{\theta}^{(k)}) - V_N''(\hat{\theta}^{(k)})\gamma^{(k)} + \frac{1}{2}(\gamma^{(k)})^{\mathrm{T}} V_N'''(\tilde{\theta}^{(k)})\gamma^{(k)} \tag{8.11}$$

The last term in (8.11), i.e. $\frac{1}{2}(\gamma^{(k)})^{\mathrm{T}} V_N'''(\tilde{\theta}^{(k)})\gamma^{(k)}$, is written in an informal way since $V_N'''(\tilde{\theta}^{(k)})$ is a tensor. The ith term of this element is correctly given as

$$(\gamma^{(k)})^{\mathrm{T}} \frac{\partial}{\partial \theta(i)} \left[V_N''(\tilde{\theta}^{(k)}) \right] \gamma^{(k)} \tag{8.12}$$

Rearrange equation (8.11) to obtain

$$V_N''(\hat{\theta}^{(k)})\gamma^{(k)} - V_N'(\hat{\theta}^{(k)}) = \frac{1}{2}(\gamma^{(k)})^{\mathrm{T}} V_N'''(\tilde{\theta}^{(k)})\gamma^{(k)} \tag{8.13}$$

Now, pre-multiply both sides of equation (8.8) by $V_N''(\hat{\theta}^{(k)})$ and combine it with equation (8.13) through the right hand side to yield

$$V_N''(\hat{\theta}^{(k)})\gamma^{(k+1)} = V_N''(\hat{\theta}^{(k)})\gamma^{(k)} - V_N'(\hat{\theta}^{(k)}) \tag{8.14}$$

$$V_N''(\hat{\theta}^{(k)})\gamma^{(k+1)} = \frac{1}{2}(\gamma^{(k)})^{\mathrm{T}} V_N'''(\tilde{\theta}^{(k)})\gamma^{(k)} \tag{8.15}$$

or

$$\gamma^{(k+1)} = \frac{1}{2}\left[V_N''(\hat{\theta}^{(k)}) \right]^{-1} (\gamma^{(k)})^{\mathrm{T}} V_N'''(\tilde{\theta}^{(k)})\gamma^{(k)} \tag{8.16}$$

Now, define a function

$$c(\delta) = \frac{1}{2}\left[\min_{|\hat{\theta}-\theta_0| \le \delta} \left| \left[V_N''(\hat{\theta}^{(k)}) \right]^{-1} \right| \right] \left[\max_{|\hat{\theta}-\theta_0| \le \delta} \left| V_N'''(\tilde{\theta}^{(k)}) \right| \right] \qquad (\delta > 0) \tag{8.17}$$

Here the above norm is defined as the Euclidean one. By defining the minimum and maximum of a matrix and a tensor, equation (8.17) can be rewritten as follow,

$$c(\delta)=\frac{1}{2}\frac{\max_i[\lambda_{\max}V_{N_i}'''(\tilde{\theta}^{(k)})]}{\lambda_{\min}V_N''(\hat{\theta}^{(k)})}$$

where $\lambda_{\min}$ and $\lambda_{\max}$ are the minimum and maximum associated eigenvalues, respectively, and $V_{N_i}'''(\tilde{\theta}^{(k)})$ is the ith matrix of the tensor $V_N'''(\tilde{\theta}^{(k)})$. For any parameter vector $\hat{\theta}$ within distance δ of the true parameter vector θ_0, the inequality

$$\frac{1}{2}\left|\left[V_N''(\hat{\theta}^{(k)})\right]^{-1}V_N'''(\tilde{\theta}^{(k)})\right|\le c(\delta) \tag{8.18}$$

is true. Now select δ so small that $\delta c(\delta)<1$. This is possible because as $\delta\to 0$, $c(\delta)$ approaches $\frac{1}{2}\left|\left[V_N''(\theta_0)\right]^{-1}V_N'''(\theta_0)\right|$ where $V_N''(\theta_0)$ is nonsingular by assumption 7. Thus $\delta c(\delta)$ converges to zero.

Let $\rho=\delta c(\delta)$ and therefore $\rho<1$. Now, at iteration k, suppose that $\hat{\theta}^{(k)}$ lies within distance δ from θ_0. Then, from (8.18)

$$\left|\gamma^{(k)}\right|=\left|\hat{\theta}^{(k)}-\theta_0\right|\le\delta \quad \text{and} \quad \left|\tilde{\theta}^{(k)}-\theta_0\right|\le\delta \tag{8.19}$$

By the definition of $c(\delta)$, equation (8.16) becomes

$$\begin{aligned}\left|\gamma^{(k+1)}\right|&=\frac{1}{2}\left|\left[V_N''(\hat{\theta}^{(k)})\right]^{-1}(\gamma^{(k)})^{\mathrm{T}}V_N'''(\tilde{\theta}^{(k)})\gamma^{(k)}\right|\\&\le c(\delta)(\gamma^{(k)})^{\mathrm{T}}\gamma^{(k)}\le\delta c(\delta)\left|\gamma^{(k)}\right|=\rho\left|\gamma^{(k)}\right|\end{aligned} \tag{8.20}$$

Consequently, $\hat{\theta}^{(k+1)}$ is also within distance δ from θ_0 because

$$\left|\hat{\theta}^{(k+1)}-\theta_0\right|=\left|\gamma^{(k+1)}\right|\le\rho\left|\gamma^{(k)}\right|\le\left|\gamma^{(k)}\right|\le\delta \tag{8.21}$$

If the initial parameter vector $\hat{\theta}^{(0)}$ is within distance δ of θ_0, then

$$\left|\gamma^{(k)}\right| \le \rho\left|\gamma^{(k-1)}\right| \le \rho^2\left|\gamma^{(k-2)}\right| \le \ldots \le \rho^k\left|\gamma^{(0)}\right| \tag{8.22}$$

Since $0 < \rho < 1$

$$\lim_{k\to\infty} \rho^k = 0 \tag{8.23}$$

and

$$\lim_{k\to\infty} \gamma^{(k)} = \mathbf{0} \tag{8.24}$$

that is

$$\lim_{k\to\infty} \hat{\theta}^{(k)} = \theta_0 \qquad \square \tag{8.25}$$

8.5 Asymptotic Distribution of The PE Estimates [1]

Following the convergence analysis of $\theta^{(k)}$ based on the PE algorithm, this section examines the limiting distribution. Let the estimate $\hat{\theta}_N$ be a minimum point of the criterion function $V_N(\theta)$. Using assumptions 7 and 8, a Taylor's series expansion of $V_N'(\theta_N)$ around θ_0, where θ_0 is the true parameter vector, retaining only the first two terms gives

$$\begin{aligned} \mathbf{0} = V_N'(\theta_N) &\approx V_N'(\theta_0) + V_N''(\theta_0)(\hat{\theta}_N - \theta_0) \\ &\approx V_N'(\theta_0) + V_\infty''(\theta_0)(\hat{\theta}_N - \theta_0) \end{aligned} \tag{8.26}$$

The second approximation follows since $V_N''(\theta_0) \to V_\infty''(\theta_0)$ with probability 1 as $N \to \infty$. Since $\hat{\theta}_N$ converges to θ_0 as N tends to infinity, for large N the dominating term in the estimate error $\hat{\theta}_N - \theta_0$ can be written as follows

$$\sqrt{N}(\hat{\theta}_N - \theta_0) \approx -[V_\infty''(\theta_0)]^{-1}\sqrt{N}V_N'(\theta_0) \tag{8.27}$$

The matrix $V_\infty''(\theta_0)$ is deterministic and assumed to be nonsingular. However, the vector $\sqrt{N}V_N'(\theta_0)$ is random. It will be shown in the following, using Theorem C.1 (Appendix C) that $\sqrt{N}V_N'(\theta_0)$ is asymptotically Gaussian distributed with a zero mean and a covariance matrix denoted by P_{0PE}. Then (8.27) gives,

$$\sqrt{N}(\hat{\theta}_N - \theta_0) \xrightarrow{dist} N(0, P_{PE}) \tag{8.28}$$

with

$$P_{PE} = [V_\infty''(\theta_0)]^{-1} P_{0PE} [V_\infty''(\theta_0)]^{-1} \tag{8.29}$$

The matrices in (8.29) must be evaluated. For this case

$$V_N(\theta) = \frac{1}{N}\sum_{t=1}^{N} \varepsilon^2(t,\theta) \qquad V_\infty(\theta) = E\{\varepsilon^2(t,\theta)\} \tag{8.30}$$

and

$$\begin{aligned} V_N'(\theta) &= -\frac{2}{N}\sum_{t=1}^{N} \varepsilon(t,\theta)\psi_{PE}^{\mathrm{T}}(t,\theta) \\ V_N''(\theta) &= \frac{2}{N}\sum_{t=1}^{N} \psi_{PE}^{\mathrm{T}}(t,\theta)\psi_{PE}(t,\theta) + \frac{2}{N}\sum_{t=1}^{N} \varepsilon(t,\theta)\frac{\partial^2}{\partial\theta^2}\varepsilon(t,\theta) \end{aligned} \tag{8.31}$$

where $\psi_E(t,\theta) = -\left(\dfrac{\partial\varepsilon(t,\theta)}{\partial\theta}\right)^{\mathrm{T}}$

From equations (5.31) to (5.36) and by employing assumption 3, we notice that both $\psi_{PE}(t,\theta)$ and $\frac{\partial^2}{\partial\theta^2}\varepsilon(t,\theta)$ depend on the input/output data up to time $t-1$. They will therefore be independent of $e(t) \equiv \varepsilon(t,\theta_0)$, where $e(t)$ is the zero mean external white noise with variance λ^2. Thus at $\theta = \theta_0$, equation (8.31) becomes

$$\begin{aligned} V_N'(\theta_0) &= -\frac{2}{N}\sum_{t=1}^{N} e(t)\psi_{PE}^{\mathrm{T}}(t,\theta_0) \\ V_N''(\theta_0) &= 2E\{\psi_{PE}^{\mathrm{T}}(t,\theta_0)\psi_{PE}(t,\theta_0)\} \end{aligned} \tag{8.32}$$

Since $e(t)$ and $\psi_{PE}(t,\theta_0)$ are uncorrelated, the result (8.28) follows from Theorem C.1. The matrix P_{0PE} can be found from

$$\begin{aligned} P_{0PE} &= \lim_{N\to\infty} E\{NV_N'(\theta_0)[V_N'(\theta_0)]^{\mathrm{T}}\} \\ &= \lim_{N\to\infty} \frac{4}{N}\sum_{t=1}^{N}\sum_{s=1}^{N} E\{e(t)\psi_{PE}^{\mathrm{T}}(t,\theta_0)e(s)\psi_{PE}(s,\theta_0)\} \end{aligned} \tag{8.33}$$

To evaluate this limit, note that $e(t)$ is white noise, $e(t)$ is independent of $e(s)$ for all $s \neq t$ and also independent of $\psi(t,\theta_0)$ for $s \leq t$. Therefore

$$\begin{aligned} P_{0PE} &= \lim_{N\to\infty}\left\{\frac{4}{N}\sum_{t=1}^{N}\sum_{s=1}^{t-1} E\{e(t)\}E\{\psi_{PE}^{\mathrm{T}}(t,\theta_0)e(s)\psi_{PE}(s,\theta_0)\}\right. \\ &\quad + \frac{4}{N}\sum_{t=1}^{N}\sum_{s=t+1}^{N} E\{e(t)\psi_{PE}^{\mathrm{T}}(t,\theta_0)\psi_{PE}(s,\theta_0)\}E\{e(s)\} \\ &\quad \left. + \frac{4}{N}\sum_{t=1}^{N}\sum_{s=t}^{t} E\{e(t)\psi_{PE}^{\mathrm{T}}(t,\theta_0)e(s)\psi_{PE}(s,\theta_0)\}\right\} \\ &= \lim_{N\to\infty} \frac{4}{N}\sum_{\substack{t=1 \\ N\to\infty}}^{N} E\{e^2(t)\}\,E\{\psi_{PE}^{\mathrm{T}}(t,\theta_0)\psi_{PE}(t,\theta_0)\} \\ &= 4\lambda^2 E\{\psi_{PE}^{\mathrm{T}}(t,\theta_0)\psi_{PE}(t,\theta_0)\} \end{aligned} \tag{8.34}$$

where

$$E\{e^2(t)\} = \lambda^2$$

Finally, from (8.29), (8.32), and (8.34) the asymptotic covariance matrix for the PE method is

$$P_{PE} = \lambda^2[E\{\psi_{PE}^{\mathrm{T}}(t,\theta_0)\psi_{PE}(t,\theta_0)\}]^{-1} \tag{8.35}$$

A reasonable estimate of P_{PE} can be found as

$$\hat{P}_{PE} = \hat{\lambda}^2\left[\frac{1}{N}\sum_{t=1}^{N}\psi_{PE}^{\mathrm{T}}(t,\hat{\theta}_N)\psi_{PE}(t,\hat{\theta}_N)\right]^{-1}$$

This means that the accuracy of $\hat{\theta}_N$ can be estimated from the data. □

8.6 Convergence Analysis Based on Recursive Identification Algorithms [1,8]

In this section, we try to extend the convergence analysis for recursive algorithms proposed by Ljung [8] to the general Wiener-Hammerstein parameter recursive estimation algorithm. To start with, consider the following recursive algorithm

$$\begin{aligned}
&\hat{\theta}(t) = \hat{\theta}(t-1) + K(t)\eta(t)\\
&K(t) = P(t)\psi^{\mathrm{T}}(t)\\
&P(t) = \frac{1}{\lambda}\left\{P(t-1) - \frac{P(t-1)\psi^{\mathrm{T}}(t)\psi(t)P(t-1)}{\psi(t)P(t-1)\psi^{\mathrm{T}}(t) + \lambda}\right\}
\end{aligned} \tag{8.36}$$

where and $\eta(t)$ can be $e_{OE}(t)$, $e_{LS}(t)$, $e_{IV}(t)$, or $\varepsilon(t)$. Algorithm (8.36) may represent any of the earlier recursive algorithms given in Chapter 7. Assume that the forgetting factor λ is time varying and set $\lambda = \lambda(t)$. Suppose that $\lambda(t)$ approaches to 1 as t

increases. The above algorithm (8.36) with λ replaced by $\lambda(t)$ corresponds to minimization of the criterion

$$V_t(\theta) = \sum_{s=1}^{t}\left[\prod_{k=s+1}^{t}\lambda(k)\right]\eta^2(s,\theta) \tag{8.37}$$

with the convention $\prod_{k=t+1}^{t}\lambda(k) = 1$.

Now, introduce the 'step length' or 'gain sequence' $\gamma(t)$ through

$$\gamma(t) = \frac{\gamma(t-1)}{\lambda(t)+\gamma(t-1)} \tag{8.38}$$

where $\gamma(1) = 1$. When $\lambda(t) \equiv 1$, we have $\gamma(t) = 1/t$. Next define the matrix $\tilde{R}(t)$ as

$$\tilde{R}(t) = \gamma(t)P^{-1}(t) \tag{8.39}$$

The covariance matrix $P(t)$ will usually be decreasing. If $\lambda(t) \equiv 1$ it will behave as $1/t$ for large t [1]. Now the algorithm (8.36) can be written as

$$\hat{\theta}(t) = \hat{\theta}(t-1) + \gamma(t)\tilde{R}^{-1}(t)\psi^{\mathrm{T}}(t)\eta(t) \tag{8.40}$$

Moreover,

$$P^{-1}(t) = \lambda(t)P^{-1}(t-1) + \psi^{\mathrm{T}}(t)\psi(t) \tag{8.41}$$

Using the substitutions (8.38) and (8.39), (8.41) thus becomes

$$\begin{aligned}\tilde{R}(t) &= \gamma(t)\lambda(t)\frac{1}{\gamma(t-1)}\tilde{R}(t-1) + \gamma(t)\psi^{\mathrm{T}}(t)\psi(t)\\ &= \tilde{R}(t-1) + \gamma(t)[\psi^{\mathrm{T}}(t)\psi(t) - \tilde{R}(t-1)]\end{aligned} \tag{8.42}$$

Both equations (8.40) and (8.42) can have the form

$$x(t) = x(t-1) + \gamma(t)\zeta(t) \tag{8.43}$$

where $\zeta(t)$ is a correction term. Through iterating (8.43), we obtain the following recursion

$$x(t) = x(0) + \sum_{k=1}^{t} \gamma(k)\zeta(k) \tag{8.44}$$

Note that $\sum_{k=1}^{t} \gamma(k)$ diverges. (Since $\gamma(k) = 1/k$ asymptotically, $\sum_{k=1}^{t} \gamma(k) \approx \log(t)$). So for (8.43) and (8.44) to converge we must have

$$E\{\zeta(t)\} = 0 \tag{8.45}$$

which is to be evaluated for a constant argument θ corresponding to the convergence point. In order to apply this idea to the algorithm (8.40) and (8.42) for a parameter vector θ, we first introduce the vector $f(\theta)$ and the matrix $G(\theta)$ by

$$\begin{aligned} f(\theta) &= E\{\eta(t,\theta)\psi^{\mathrm{T}}(t,\theta)\} \\ G(\theta) &= E\{\psi^{\mathrm{T}}(t,\theta)\psi(t,\theta)\} \end{aligned} \tag{8.46}$$

8.6.1 Associated differential equations

If we assume that $\hat{\theta}(t)$ in (8.40) converges to θ^* and $\tilde{R}(t)$ in (8.42) to R^*, then through the expectation (8.45) to the algorithm (8.40) and (8.42) we have

$$\begin{aligned} (R^*)^{-1} f(\theta) &= \mathbf{0} \\ G(\theta) - R^* &= \mathbf{0} \end{aligned} \tag{8.47}$$

From (8.47), possible limit points of (8.40) must satisfy

$$f(\theta^*) = \mathbf{0} \tag{8.48}$$

In order to proceed, consider the following set of ordinary differential equations (ODE)

$$\begin{aligned}\frac{d}{d\tau}\theta_\tau &= R_\tau^{-1} f(\theta_\tau) \\ \frac{d}{d\tau}R_\tau &= G(\theta_\tau) - R_\tau\end{aligned} \tag{8.49}$$

If the ODEs (8.49) are solved numerically by the Euler method and computed for $\tau = t_1, t_2, \cdots$. Then

$$\begin{aligned}\theta_{t_2} &\approx \theta_{t_1} + (t_2 - t_1) R_{t_1}^{-1} f(\theta_{t_1}) \\ R_{t_2} &\approx R_{t_1} + (t_2 - t_1)[G(\theta_{t_1}) - R_{t_1}]\end{aligned} \tag{8.50}$$

Note the similarity between (8.50) and the algorithm (8.40) and (8.42). This similarity suggests that the solutions to the deterministic ODE will be close to the paths corresponding to the algorithm (8.36) if the ODE is used with the following time scale

$$\tau = \sum_{k=1}^{t} \gamma(k) \approx \log(t) \tag{8.51}$$

Through the above heuristic analysis, the possible limit points of $\hat{\theta}(t)$ as t tends to infinity are the stable stationary points of the differential equation (8.49), the estimates $\hat{\theta}(t)$ converge *locally* to stable stationary points. If $\hat{\theta}(t_0)$ is close to a stable stationary point θ^* and the gain $K(t)$, $t \geq t_0$ are small enough, then $\hat{\theta}(t)$ converges to θ^*.

Next we examine the (locally) stable stationary points of (8.49), which constitute the possible limits of $\hat{\theta}(t)$. Let θ^*, $R^* = G(\theta^*)$ be a stationary point of the ODE (8.49). Now, set $r^* = \mathrm{vec}(R^*), r_\tau = \mathrm{vec}(R_\tau)$, see Definition C.1. If the ODE is linearized around (θ^*, R^*), then

$$\frac{d}{d\tau}\begin{pmatrix}\theta_\tau-\theta^*\\ r_\tau-r^*\end{pmatrix}=\begin{pmatrix}(R^*)^{-1}\left.\dfrac{\partial f(\theta)}{\partial\theta}\right|_{\theta=\theta^*} & \mathbf{0}\\ \mathbf{X} & -\mathbf{I}\end{pmatrix}\begin{pmatrix}\theta_\tau-\theta^*\\ r_\tau-r^*\end{pmatrix} \tag{8.52}$$

In (8.52) the block marked $\mathbf{X}$ is apparently of no importance for the local stability properties. The stationary point $\{\theta^*, R^*\}$ will be stable if and only if the matrix

$$L(\theta^*)=[G(\theta^*)]^{-1}\left.\frac{\partial f(\theta)}{\partial\theta}\right|_{\theta=\theta^*} \tag{8.53}$$

has all eigenvalues in the strict left hand-plane.

8.6.2 Convergence analysis of the RPE method

We now consider the possible stationary points of (8.49) when the algorithm (8.36) is replaced by the RPE algorithm, i.e. equation (7.18). Then

$$\begin{aligned}\mathbf{0}=f(\theta)&=E\{\varepsilon(t,\theta)\psi_{PE}^{\mathrm{T}}(t,\theta)\}\\ &=-\frac{1}{2}\frac{\partial}{\partial\theta}E\{\varepsilon^2(t,\theta)\}\end{aligned} \tag{8.54}$$

Hence only the stationary points of the asymptotic criterion function

$$V_\infty(\theta)=E\{\varepsilon^2(t,\theta)\} \tag{8.55}$$

can be possible convergence points of $\hat{\theta}(t)$. If $V_\infty(\theta)$ has a unique minimum at $\theta=\theta_0$, i.e. assumption 8, then convergence can take place to the true parameters only. Further, for the true parameter vector

$$\begin{aligned}\left.\frac{\partial f(\theta)}{\partial\theta}\right|_{\theta=\theta_0}&=\left.\frac{\partial}{\partial\theta}\left[E\{\psi_{PE}^{\mathrm{T}}(t,\theta)\varepsilon(t,\theta)\}\right]\right|_{\theta=\theta_0}\\ &=E\left\{\left.\frac{\partial\psi_{PE}^{\mathrm{T}}}{\partial\theta}(t,\theta)\right|_{\theta=\theta_0}\varepsilon(t,\theta_0)\right\}+E\left\{\psi_{PE}^{\mathrm{T}}(t,\theta_0)\left.\frac{\partial\varepsilon(t,\theta)}{\partial\theta}\right|_{\theta=\theta_0}\right\}\\ &=-E\{\psi_{PE}^{\mathrm{T}}(t,\theta_0)\psi_{PE}(t,\theta_0)\}=-G(\theta_0)\end{aligned} \tag{8.56}$$

Thus

$$L(\theta_0) = -\mathbf{I} \tag{8.57}$$

which has all eigenvalues in the left half-plane.

Chapter 9

SIMULATION RESULTS

9.1 Introduction

A number of identification algorithms for the Wiener-Hammerstein model representing nonlinear dynamic systems have been presented so far. The basic offline techniques available for the identification of linear system models have been extended, and their recursive counterparts have been derived as well. In addition, a genetic algorithm has been developed in order to estimate the Wiener-Hammerstein model parameters without incorporating any parameter encoding techniques.

As has been stated earlier, in order to verify the feasibility of the proposed algorithms, *analysis* and/or *simulation* should be carried out. Chapter 8 has been concerned with the analysis, and here we complement the work by presenting some results based on simulation.

This chapter consists of the following. Section 9.2 presents two Wiener-Hammerstein models that have been used in our simulation study. Simulation results are presented for different identification algorithms, for both offline, including the GAs, and the recursive ones. The results are analyzed, compared and possible explanations for the observations are made.

Section 9.3 illustrates the Wiener-Hammerstein model identification of a nonlinear DC generator, which mimics a real-world estimation problem.

9.2 Parameter Estimation via Simulation

Consider the general Wiener-Hammerstein nonlinear model given in Figure 9.1.

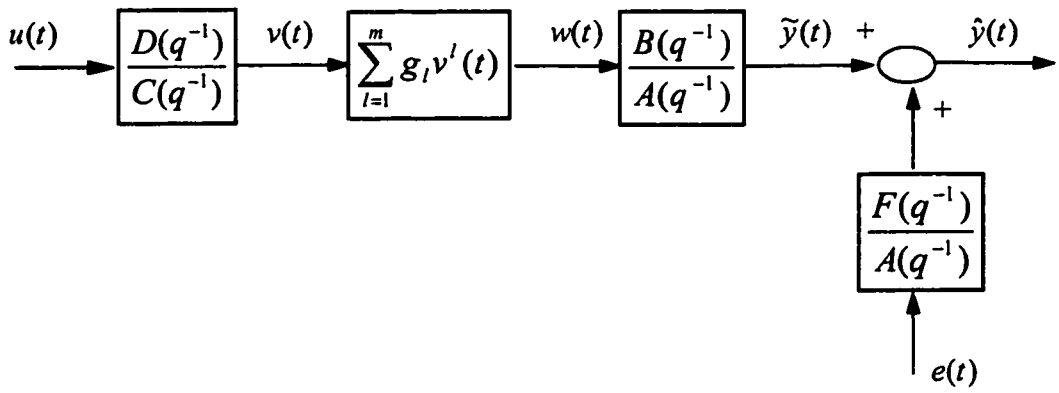


Figure 9.1. A general Wiener-Hammerstein model.

Different input signals $u(t)$ have been used in our simulation studies. Namely,

- A Gaussian distributed input with a zero-mean and variance of 0.5.
- A uniformly distributed input with $-1 \le u(t) \le 1$.
- A PRBS input signal shifting between -1 and 1.
- A 4-level Modified Pseudo-Random Multilevel Signal (MPRMS) input signal with levels at $-1.00, -0.33, 0.33$ and 1.00. While the minimum and maximum switching times have been 3 and 10 samples respectively, the levels and the time samples had equal switching probabilities.

$e(t)$ is considered to be independent, identically distributed zero-mean Gaussian sequence. The variance of $e(t)$, i.e. λ^2, can be varied in simulation to produce certain Noise-to-Signal Ratio (NSR) as defined below

$$\text{NSR} = \frac{\lambda^2}{\text{variance}(\tilde{y})} \times 100$$

where $\tilde{y}(t)$ is the noise-free model output, see Figure 9.1. This definition of NSR has been used in the engineering literature.

Example 9.1: A Wiener-Hammerstein system

For this example, the Wiener-Hammerstein model defining polynomials are given as

$$A(q^{-1}) = 1 + a_1 q^{-1} + a_2 q^{-2}$$
$$B(q^{-1}) = 1 + b_1 q^{-1}$$
$$C(q^{-1}) = 1 + c_1 q^{-1} + c_2 q^{-2}$$
$$D(q^{-1}) = 1 + d_1 q^{-1}$$
$$F(q^{-1}) = 1 + f_1 q^{-1}$$

$$w(t) = g_1 v(t) + g_2 v^2(t) + g_3 v^3(t)$$

where the actual parameter values are given as follows

$$[a_1, a_2, b_1, c_1, c_2, d_1, f, g_1, g_2, g_3]^T = [-0.4, 0.25, 0.3, 0.5, -0.25, 0.7, 0.6, 0.5, -1.8, 2.5]^T$$

Example 9.2: Another Wiener-Hammerstein system

The polynomials of the Wiener-Hammerstein model are given as

$$A(q^{-1}) = 1 + a_1 q^{-1} + a_2 q^{-2}$$
$$B(q^{-1}) = 1 + b_1 q^{-1}$$
$$C(q^{-1}) = 1 + c_1 q^{-1} + c_2 q^{-2} + c_3 q^{-3}$$
$$D(q^{-1}) = 1 + d_1 q^{-1} + d_2 q^{-2}$$
$$F(q^{-1}) = 1 + f_1 q^{-1} + f_2 q^{-2}$$

$$w(t) = g_1 v(t) + g_3 v^3(t)$$

and the actual parameter values are taken as

$$[a_1, a_2, b_1, c_1, c_2, c_3, d_1, d_2, f_1, f_2, g_1, g_3]^T =$$
$$[-0.4, 0.25, 0.3, -0.6, -0.25, 0.15, -0.55, -0.2, 0.6, -0.1, 1.0, 2.5]^T$$

For Example 9.1, the number of samples, the maximum number of Gauss-Newton iterations, and the maximum number of GAs generations for each algorithm are given in Table 9.1. While $\hat{\theta}^{(0)}$ for Gauss-Newton algorithms is chosen very close to zero, the initial estimates $\hat{\theta}_0$, P_0 and λ for the recursive algorithms have been taken as

$\hat{\theta}_0 = \mathbf{0}$; $P_0 = 10^{3}\mathbf{I}$; $\lambda = 0.99$

Whereas for the Stochastic Approximation (SA) algorithm, A suitable choice of $\gamma(t)$ is $Ct^{-\alpha}$ with $0.5 < \alpha \leq 1$ and C a positive constant. In this research, $\alpha = 0.75$ and $C = 0.001$.[3,8]

Algorithm	no. samples	no. iterations	no. generations
OE	1000	50	-
PE	1000	50	-
NLS	1000	50	-
IV	1000	50	-
GAs	100	-	200
ROE	2000	-	-
RPE	2000	-	-
RNLS	2000	-	-
RIV	2000	-	-
SA	6000	-	-

Table 9.1. Number of samples, iterations, and generations used for Example 9.1.

The Genetic Algorithms used for identification herein are based on the real values of the parameters as they are, with no coding. A population P of a fixed size 300 is initialized randomly. Only 25% of P has been selected for reproduction. The perturbation probability P_p, the crossover probability P_c, and the mutation probability P_m are taken as 0.05, 0.9 and 0.3, respectively. The parameter spaces of the three

linear systems are chosen so that the poles and zeros are inside the unit circle. While the nonlinearity parameters are chosen to fall in the space of -5 to 5.

The following tables, Table 9.2 to Table 9.5, compare the estimates of the above parameters based on the proposed offline and online algorithms. The numerical results are obtained from Monte Carlo simulation of 30 experiments where different realizations of u and e are generated. Two different NSR are incorporated, i.e. 10% and 50%. In order to facilitate a quick comparison the Mean Normalized Error (MNE) is included. The MNE is defined as

$$\text{MNE} = \frac{\left\|\hat{\theta}_{mean} - \theta_0\right\|^2}{\left\|\theta_0\right\|^2}$$

where $\hat{\theta}_{mean}$ is the mean value of the estimates over the 30 experiments conducted. While the MNE is a composite measure of bias in the parameters, the Average Standard Deviation (ASD) is a measure in their variance. The ASD is defined as below

$$\text{ASD} = \frac{\sum_{i=1}^{n_\theta} \sigma_i}{n_\theta}$$

where σ_i is the ith parameter standard deviation over the 30 runs. The tables also include the Average Iteration For Convergence (AIFC). For offline algorithms, convergence is assumed when each of the relative changes in the parameter estimate in successive iteration has been less than 2%.

For a Gaussian input and a 10% NSR, Table 9.6 shows a comparison of computational as well as time requirements of the proposed methods based on

Example 9.1. The table includes the time and the number of computations per iteration as well as the average total time and computations required until convergence. A personal computer with an Intel 266MHz Pentium II Processor was used for the simulation via MATLAB.

Input Signal	NSR	Algorithm	MNE	ASD	AIFC
Gaussian input	10%	OE	0.1508	0.0604	20
		PE	0.0112	0.0024	10
		NLS	0.1204	0.0030	10
		IV	0.0686	0.0530	16
		GAs	0.0609	0.0095	160
	50%	OE	0.3047	0.2576	23
		PE	0.0413	0.0082	10
		NLS	0.1970	0.0616	9
		IV	0.1119	0.1388	13
		GAs	0.0561	0.0467	163
Uniform input	10%	OE	0.1367	0.0960	30
		PE	0.0172	9.0139E-4	14
		NLS	0.1355	0.0011	7
		IV	0.1274	0.0778	8
		GAs	0.0560	0.0155	165
	50%	OE	0.1270	0.0427	27
		PE	0.0502	0.0197	9
		NLS	0.1873	0.0121	11
		IV	0.1440	0.1033	12
		GAs	0.0625	0.0347	170
PRBS input	10%	OE	0.3961	0.2232	44
		PE	0.0709	0.0893	16
		NLS	0.2880	0.0250	10
		IV	0.1864	0.2152	33
		GAs	0.0602	0.0376	180
	50%	OE	0.9028	0.2513	45
		PE	0.1175	0.0975	19
		NLS	0.3003	0.0355	16
		IV	0.2511	0.1024	35
		GAs	0.0736	0.0705	165
4-Level MPRMS input	10%	OE	0.1505	0.1200	13
		PE	0.0497	0.0518	15
		NLS	0.2012	0.1107	16
		IV	0.0818	0.0948	30
		GAs	0.0897	0.0684	161
	50%	OE	0.1863	0.1248	21
		PE	0.0603	0.0095	17
		NLS	0.2880	0.0636	11
		IV	0.1632	0.0915	35
		GAs	0.0975	0.0846	152

Table 9.2. Summary of offline identification using Example 9.1.

Input Signal	NSR	Algorithm	MNE	ASD	AIFC
Gaussian input	10%	ROE	0.0638	0.0281	620
		RPE	0.0281	0.0169	795
		RNLS	0.1264	0.0152	695
		RIV	0.0430	0.0267	935
		SA	0.4392	0.0040	6000
	50%	ROE	0.1100	0.0955	790
		RPE	0.1591	0.0134	1425
		RNLS	0.2637	0.0313	537
		RIV	0.2277	0.0518	928
		SA	0.4337	0.0220	6000
Uniform input	10%	ROE	0.0840	0.0461	880
		RPE	0.0217	0.0157	1165
		RNLS	0.1233	0.0281	835
		RIV	0.0351	0.0252	1300
		SA	0.8510	3.0988E-4	6000
	50%	ROE	0.1684	0.2065	985
		RPE	0.1430	0.1755	1250
		RNLS	0.1693	0.1831	870
		RIV	0.1010	0.1027	1635
		SA	0.8334	7.2331E-4	6000
PRBS input	10%	ROE	0.3077	0.2662	1185
		RPE	0.1933	0.0190	1563
		RNLS	0.2251	0.2135	1250
		RIV	0.2035	0.2349	1516
		SA	1.3169	0.0962	6000
	50%	ROE	1.4431	0.2970	1880
		RPE	0.3969	0.0536	1741
		RNLS	0.8650	0.1590	934
		RIV	0.7091	0.2014	1550
		SA	0.8361	0.0398	6000
4-Level MPRMS input	10%	ROE	0.1202	0.1599	1130
		RPE	0.0811	0.0189	1250
		RNLS	0.1414	0.0488	955
		RIV	0.1071	0.1324	1300
		SA	0.7436	0.0164	6000
	50%	ROE	0.2894	0.2557	1330
		RPE	0.2164	0.0858	1205
		RNLS	0.3456	0.1576	980
		RIV	0.5954	0.2047	1825
		SA	0.7492	0.0058	6000

Table 9.3. Summary of online identification using Example 9.1.

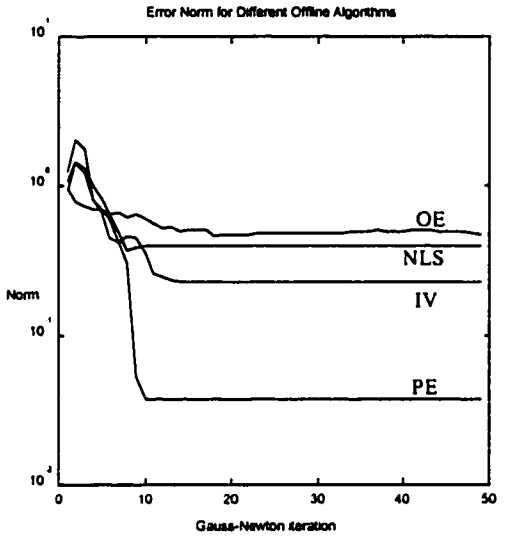


Figure 9.2. Error Norm for different offline algorithms for Example 9.1 with NSR 10% and Gaussian input.

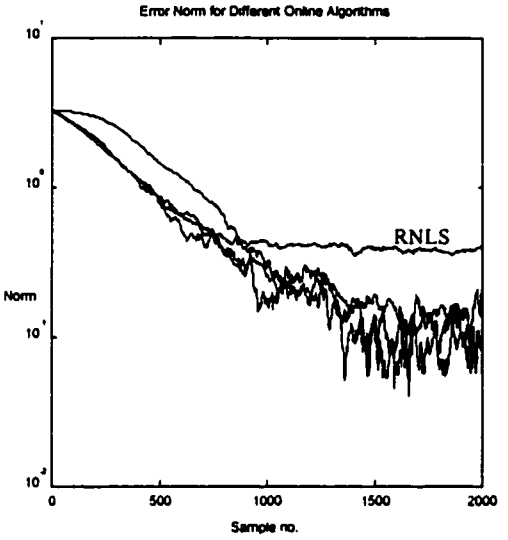


Figure 9.3. Error Norm for different online algorithms for Example 9.1 with NSR 10% and Gaussian input.

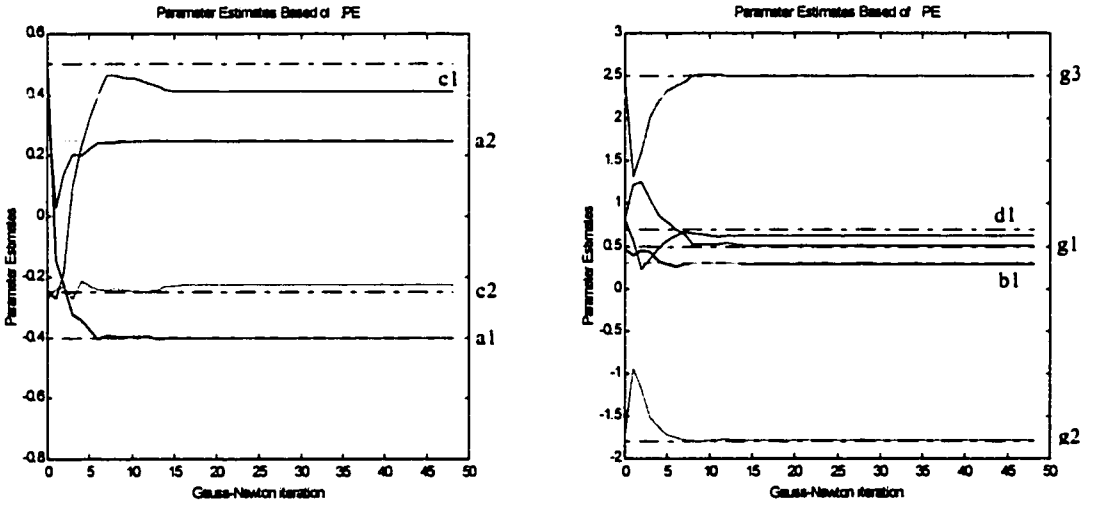


Figure 9.4. Parameter estimates based on PE for Example 9.1 with NSR 10% and Gaussian input.

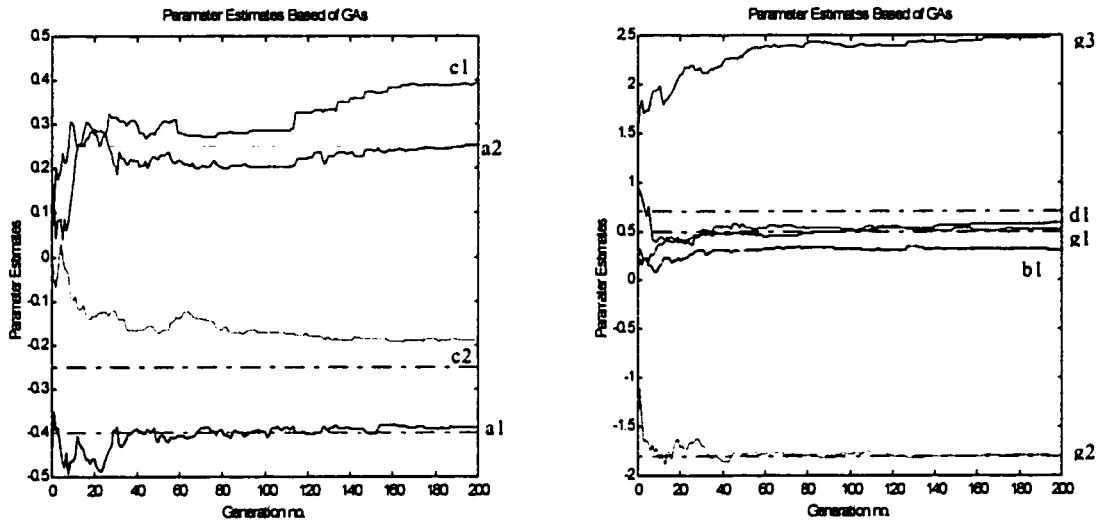


Figure 9.5. Parameter estimates based on GAs for Example 9.1 with NSR 10% and Gaussian input.

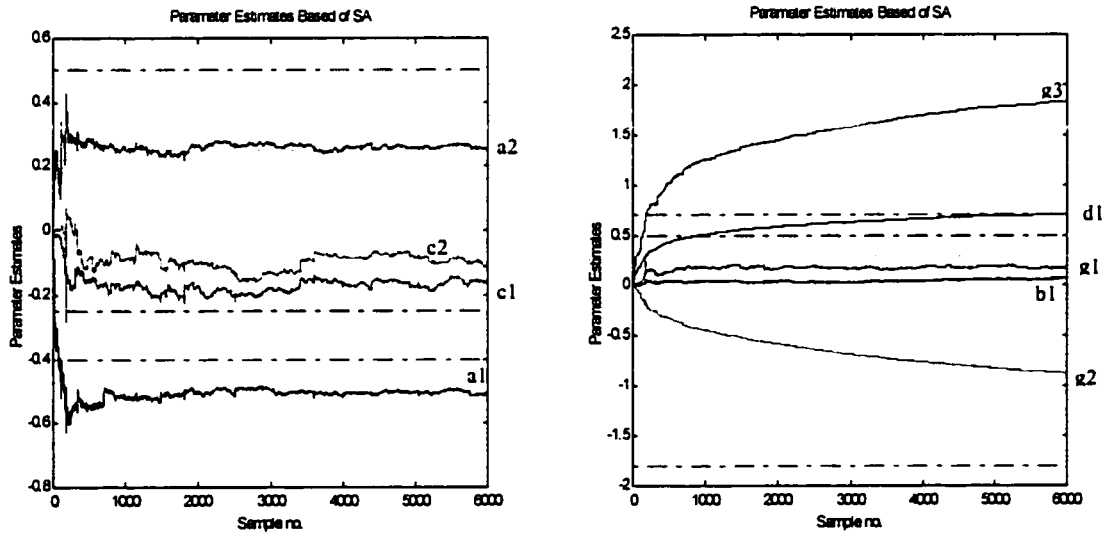


Figure 9.6. Parameter estimates based on SA for Example 9.1 with NSR 10% and Gaussian input.

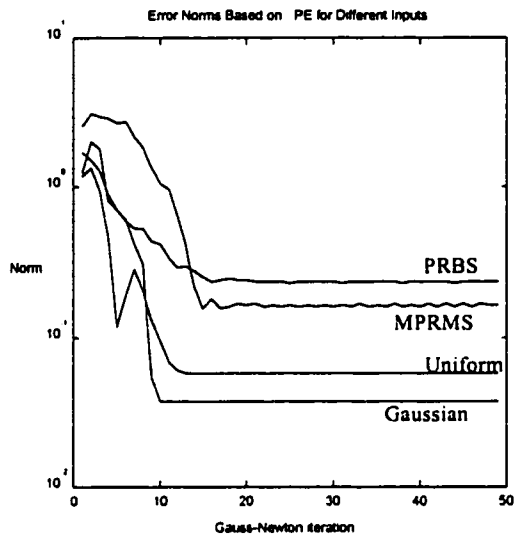


Figure 9.7. Error Norms based on PE due to different inputs for Example 9.1 with NSR 10%.

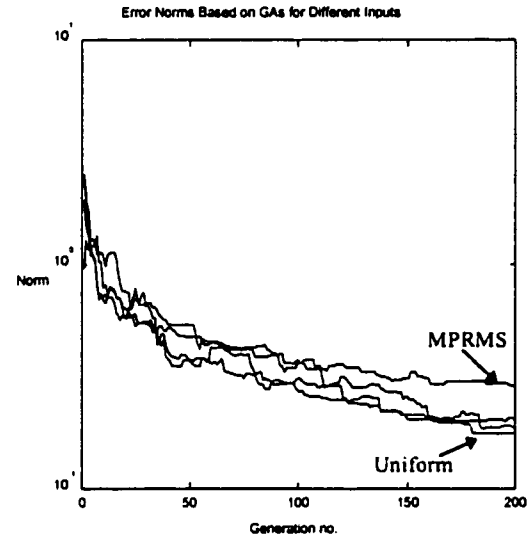


Figure 9.8. Error Norms based on GAs due to different inputs for Example 9.1 with NSR 10%.

Input Signal	NSR	Algorithm	MNE	ASD	AIFC
Gaussian input	10%	OE	0.1064	0.0708	42
		PE	0.0042	0.0038	9
		NLS	0.1492	0.0103	13
		IV	0.0289	0.0047	26
		GAs	0.1424	0.0279	152
	50%	OE	0.1288	0.0428	22
		PE	0.0600	0.0400	11
		NLS	0.2230	0.0860	9
		IV	0.1129	0.1479	19
		GAs	0.1165	0.0288	145
Uniform input	10%	OE	0.0838	0.0573	17
		PE	0.0062	0.0033	9
		NLS	0.1671	0.0187	23
		IV	0.0502	0.0155	13
		GAs	0.1294	0.0824	147
	50%	OE	0.1381	0.1115	23
		PE	0.0375	0.0185	15
		NLS	0.1991	0.0723	14
		IV	0.0800	0.0443	17
		GAs	0.0977	0.0983	174
PRBS input	10%	OE	1.0420	0.2016	46
		PE	0.3420	0.0106	16
		NLS	1.1188	0.0588	21
		IV	0.4470	0.1964	50
		GAs	0.1742	0.0547	161
	50%	OE	1.1322	0.2648	50
		PE	0.3932	0.0346	50
		NLS	1.1427	0.0742	20
		IV	0.4691	0.1959	37
		GAs	0.1931	0.0719	155
4-Level MPRMS input	10%	OE	0.0960	0.0592	35
		PE	0.0269	0.0112	47
		NLS	0.1984	0.0517	17
		IV	0.0523	0.0513	37
		GAs	0.1652	0.0317	165
	50%	OE	0.1495	0.2842	26
		PE	0.0552	0.0915	13
		NLS	0.2292	0.1410	18
		IV	0.1399	0.2393	35
		GAs	0.1712	0.0459	165

Table 9.4. Summary of offline identification using Example 9.2.

Input Signal	NSR	Algorithm	MNE	ASD	AIFC
Gaussian input	10%	ROE	0.0993	0.0551	2755
		RPE	0.0150	0.0197	2453
		RNLS	0.1597	0.0598	929
		RIV	0.0615	0.0429	2414
		SA	0.3834	0.1518	9000
	50%	ROE	0.1287	0.1984	2180
		RPE	0.0497	0.0771	1760
		RNLS	0.2574	0.9387	2050
		RIV	0.1844	0.1865	2375
		SA	0.3990	0.0215	9000
Uniform input	10%	ROE	0.1039	0.0517	1545
		RPE	0.0233	0.0185	1436
		RNLS	0.1521	0.0449	977
		RIV	0.0507	0.0494	1740
		SA	0.7578	7.5368e-4	9000
	50%	ROE	0.2156	0.2027	2160
		RPE	0.1061	0.0903	1240
		RNLS	0.2369	0.1274	1780
		RIV	0.4140	0.1729	2750
		SA	0.7471	5.5345e-4	9000
PRBS input	10%	ROE	1.0521	0.0894	2120
		RPE	0.7898	0.0913	1420
		RNLS	1.2032	0.1582	1300
		RIV	1.0752	0.1485	1630
		SA	0.5993	0.0317	9000
	50%	ROE	1.3521	0.2017	1060
		RPE	1.1136	0.0863	1830
		RNLS	1.4995	0.1778	1140
		RIV	0.9801	0.1811	2235
		SA	0.6440	0.0362	9000
4-Level MPRMS input	10%	ROE	0.2910	0.0779	2190
		RPE	0.0419	0.0450	1120
		RNLS	0.2196	0.0980	1025
		RIV	0.2032	0.0498	1705
		SA	0.8257	4.1920e-4	9000
	50%	ROE	0.9454	0.2826	2365
		RPE	0.2448	0.0295	1390
		RNLS	0.7955	0.0957	780
		RIV	0.5665	0.1898	2540
		SA	0.7349	0.0022	9000

Table 9.5. Summary of online identification using Example 9.2.

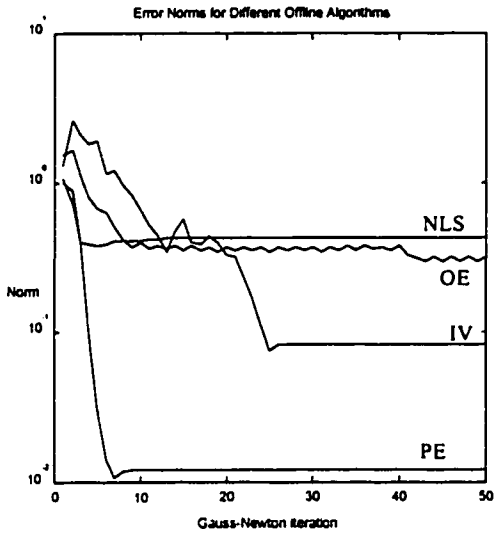


Figure 9.9. Error Norms for different offline algorithms for Example 9.2 with NSR 10% and Gaussian input.

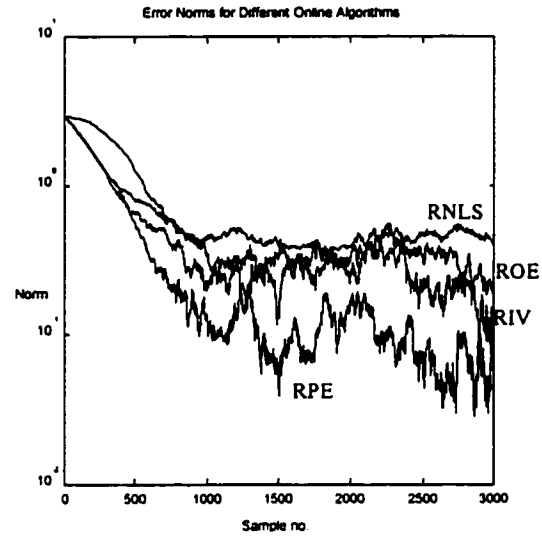
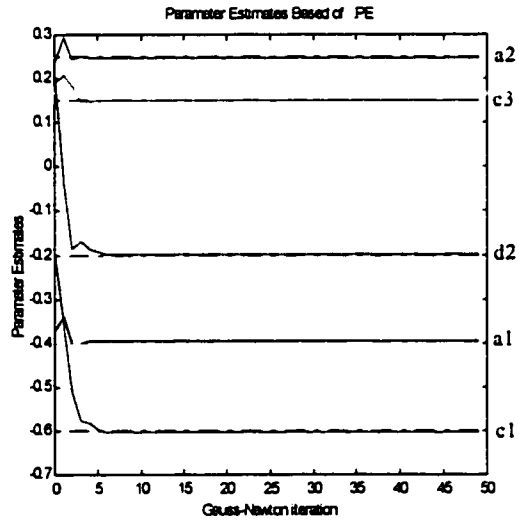


Figure 9.10. Error Norms for different online algorithms for Example 9.2 with NSR 10% and Gaussian input.



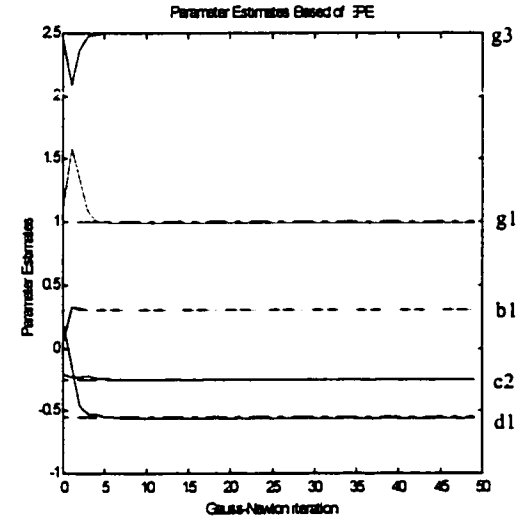


Figure 9.11. Parameter estimates based on PE for Example 9.2 with NSR 10% and Gaussian input.

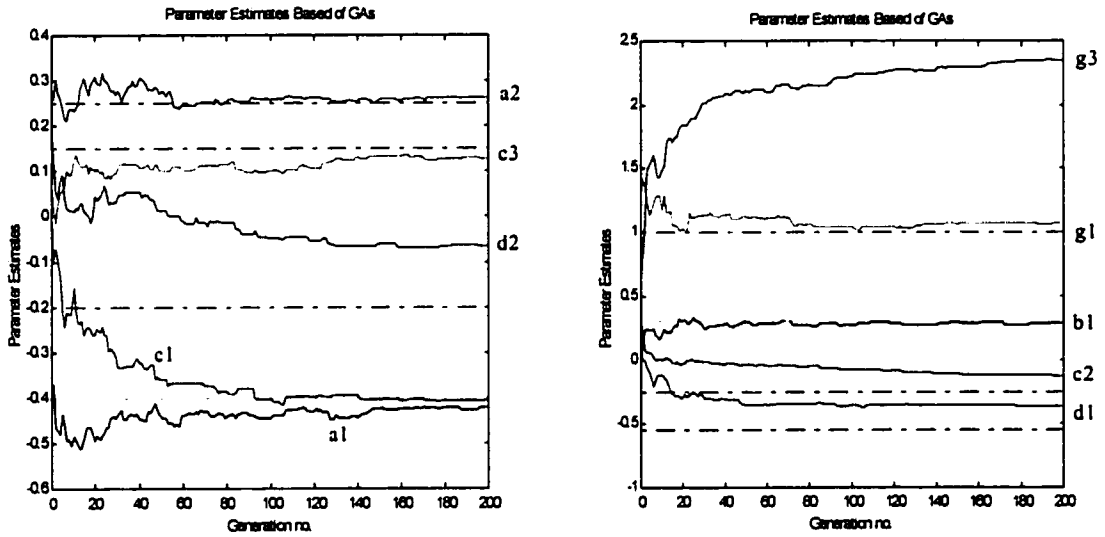


Figure 9.12. Parameter estimates based on GAs for Example 9.2 with NSR 10% and Gaussian input.

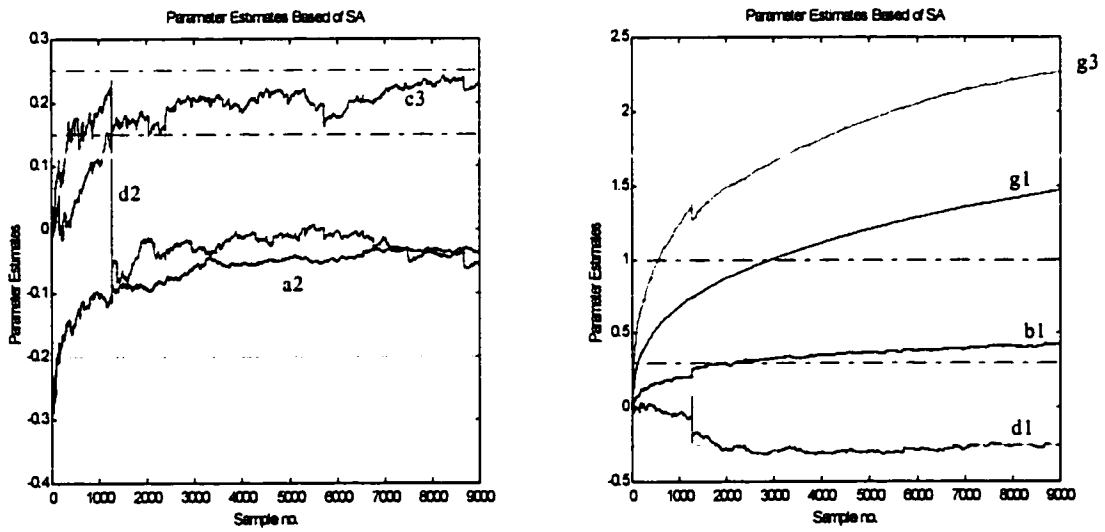


Figure 9.13. Parameter estimates based on SA for Example 9.2 with NSR 10% and Gaussian input.

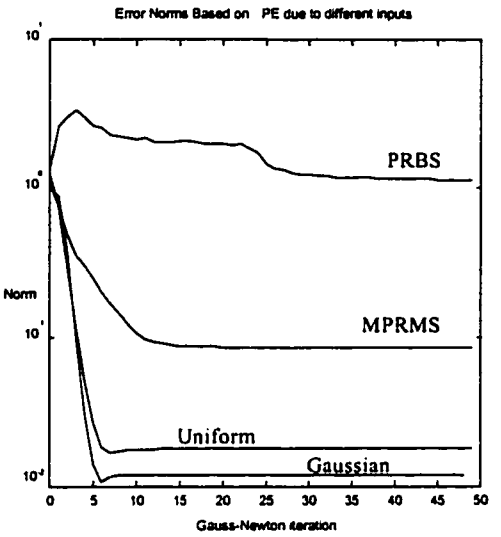


Figure 9.14. Error Norms based on PE due to different inputs for Example 9.2 with NSR 10%.

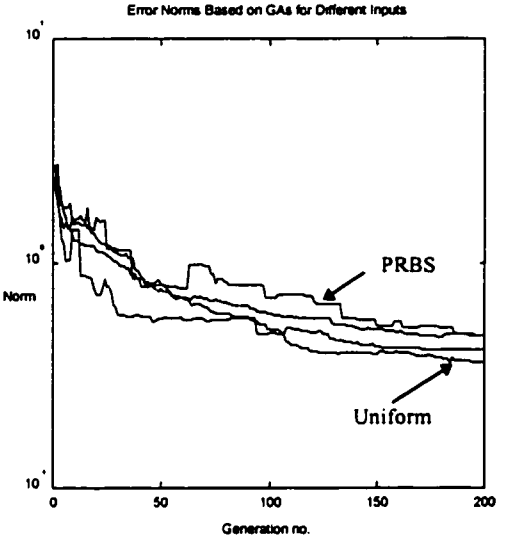


Figure 9.15. Error Norms based on GAs due to different inputs for Example 9.2 with NSR 10%.

Algorithm	Computations and time per iteration		AIFC for 10% NSR	Total requirements until convergence (computations and time)	
	total op. counts[*]	*time (sec)*		*total op. counts*	*time (sec)*
OE	153629	2.91	20	3072580	58.2
PE	139690	2.75	10	1396900	27.5
NLS	113748	2.47	10	1137480	24.7
IV	181544	4.07	16	2904704	65.12
GAs	672143	9.23	160	107542880	1476.8
ROE	438	1.65	620	271560	1023
RPE	458	1.65	935	428230	1542.75
RNLS	421	1.59	695	292595	1105.05
RIV	492	1.65	795	391140	1311.75
SA	387	1.59	6000	2322000	9540

Table 9.6. Computation and time comparison for Example 9.1
([*] total operations count = $\sum$ addition/subtraction/multiplication/division per iteration).

As a conclusion from the above identification exercises, the following points and remarks can be made:

- *Accuracy.* By observing Tables 9.2 to 9.5 and Figures 9.2 to 9.15, it can be said that for different input signals and different NSRs the Prediction Error algorithms generally yield the most accurate estimates as reflected by the MNE, ASD and error norms, while the least accurate algorithms are the Output Error and the NLS.

  The least squares estimates have been poor due to the biasedness. The poor performance of the output error method, however, does not resemble that of the linear system identification. It appears that the poor performance of the OE is due to convergence to a local minimum. Among the four offline algorithms considered, the OE is the most nonlinear one. This high degree of nonlinearity makes it converge to a bad minimum.

  Although the recursive algorithms are approximate of their offline counterparts and they are supposed to produced worse estimates, observing the above tables indicate that they produce better estimates than the offline do. A reasonable justification is that there are several parameters to play with in the recursive algorithms, and one may claim that different choices of these parameters may result in much improved estimates.

  So, in general, the results obtained by applying the PE, OE, EE, and IV algorithms to the nonlinear Wiener-Hammerstein models go in parallel with the ones found in the literature of the linear system identification. Some inconsistency of this observation, however, may be seen in above tables, when the ROE scheme yielded better estimates than the RPE. It is our conjecture that RPE in this case failed to converge to the global minimum. All the algorithms provided in this research are nonlinear and convergence to local minima may occur.

- *The effect of input signals and NSRs.* As for the linear system identification, it can be seen from the above tables and figures that the most exciting input signal is the white noise, being either normally or uniformly distributed. The worst estimation results are due to the PRBS input. Some estimates which are due to the PRBS input have quite high MNE, which may mean that the PRBS is not rich enough for the identification of Wiener-Hammerstein model parameters. The MPRMS inputs provide slightly better estimates and hence they could be more exciting than the PRBS in the current identification problem. Both accuracy and convergence rates are affected by the increase of the NSR, which is expected and is consistent with the analysis in Section 8.5.
- *Genetic algorithms.* The adopted GAs provide somehow 'robust' estimates. The GAs are 'robust' in the sense that they provide almost the same estimates without being affected by either input type or the NSR. However, as shown in Table 9.6, the amount of time and computation is much more higher than other offline algorithms. As an example, the GAs consume almost 23 times the amount the IV needs for convergence. GAs also require the range of parameters to be known. Further, the accuracy of the parameter estimates are worse than the ones produced via PE.
- *Stochastic approximation.* The SA needs the least amount of computations and hence the least amount of time to process one iteration. In addition, one may observe from the above tables that the SA estimates are somehow similar for all the input type or the NSR. However, the price of being simple is paid in the huge number of iterations that are needed for convergence and in large parameter

variances. As shown in Figures 9.6, it took 6000 iterations for Example 9.1 while the estimates did not finely converge to the actual values.

9.3 Application to the Estimation of a Nonlinear DC Generator [52]

The proposed PE algorithm is also used for the estimation of a nonlinear DC generator parameters, see Figure 9.16.

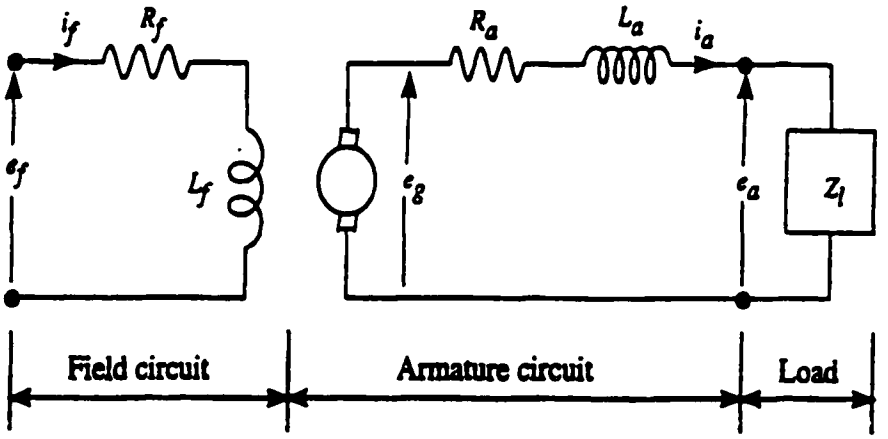


Figure 9.16. Schematic diagram of the DC generator.

The linear subsystems can be represented by the following transfer functions:

$$G_1(s) = \frac{I_f(s)}{E_f(s)} = \frac{1}{L_f s + R_f} \tag{9.1}$$

$$G_2(s) = \frac{E_a(s)}{E_g(s)} = \frac{Z_l(s)}{L_a s + R_a + Z_l(s)} \tag{9.2}$$

The various quantities are indicated in Figure 9.16. e_g represents the system nonlinearity, consisting of a dead-zone combined with a saturation curve:

$$e_g = \begin{cases} 0 & |i_f| \le \gamma_3 \\ \text{sign}(i_f).\gamma_1.\left\{1 - e^{(-\text{sign}(i_f).i_f+\gamma_3)/\gamma_2}\right\} & |i_f| > \gamma_3 \end{cases} \tag{9.3}$$

with $\text{sign}(x) \equiv +1(x>0), -1(x<0)$. γ_1, γ_2, and γ_3 are the nonlinearity parameters whose values are given in Table 9.7.

To illustrate the nonlinearity in the output, Figure 9.17 shows the frequency spectrum of e_a due to sinusoidal input $e_f(t) = \sin(0.2\pi t)$ volts.

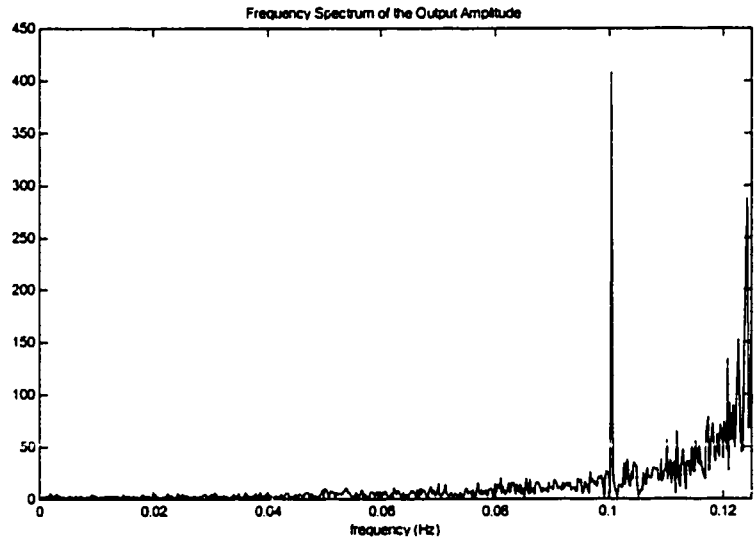


Figure 9.17. An illustration of the nonlinearity presence in the DC generator output.

Assuming that the load impedance is purely resistive, that is $Z_l = R_l$, the discretization of expressions (9.1) and (9.2) lead to the following discrete-time expressions:

$$H_1(q^{-1}) = \frac{1}{1 - c_1 q^{-1}} = \frac{1}{1 - e^{-R_f T'/L_f} q^{-1}} \tag{9.4}$$

$$H_2'(q^{-1}) = \frac{q^{-1}}{1 - a_1 q^{-1}} = \frac{q^{-1}}{1 - e^{-(R_a + R_l)T/L_a} q^{-1}} \tag{9.5}$$

where T is the sampling period. The actual parameter values of the continuous-time model are given in Table 9.7.

		Actual Values
Discrete System Parameters	a_1	-0.9179
	c_1	-0.9380
	f_1	0.5000
	γ_1'	11.5080
	γ_2'	39.1006
	γ_3'	0.0129
Physical parameters	L_f (mH)	0.5000
	L_a (mH)	0.7000
	γ_1 (V)	210.000
	γ_2 (A)	3.0303
	γ_3 (mA)	1.0000
	R_f (Ω)	0.8000
	R_a (Ω)	0.5000
	R_l (Ω)	1.0000
	T (*msec*)	0.0400

Table 9.7. True parameter values of the nonlinear DC generator problem.

Now we would like to approximate the above non-smooth nonlinearity using a power polynomial of degree n, where $n > 1$. In the following, we plot e_g' versus the input i_f for both the non-smooth nonlinearity and an approximating power polynomial of order $n = 5$:

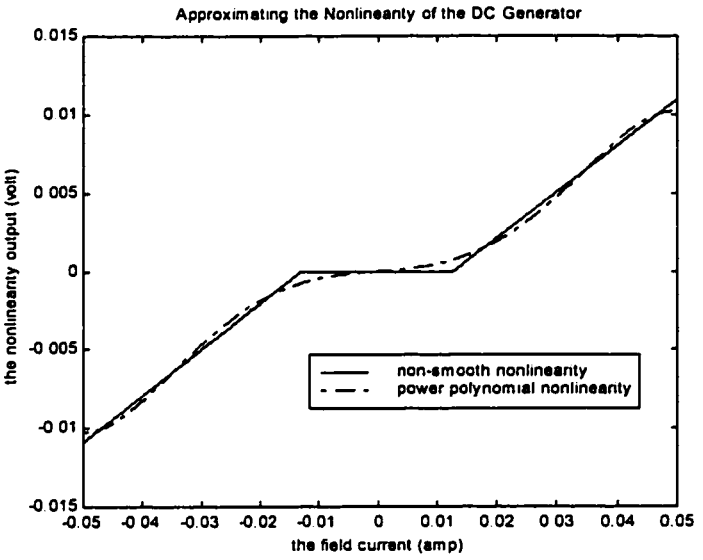


Figure 9.18. Approximating the nonlinearity in the DC generator system.

The approximating power polynomial is:

$$\hat{e}_g' = -45843.81(i_f)^5 + 185.2(i_f)^3 + 0.027881 i_f$$

The next step is to apply the PE algorithm in order to represent and estimate the system shown in Figure 9.16 by a Wiener-Hammerstein model. The actual input/output data are generated using the circuit shown in Figure 9.16 and the Wiener-Hammerstein model used in the identification had the following structure:

$$A(q^{-1}) = 1 + a_1 q^{-1} + a_2 q^{-2}$$
$$B(q^{-1}) = 1 + b_1 q^{-1}$$
$$C(q^{-1}) = 1 + c_1 q^{-1} + c_2 q^{-2}$$
$$D(q^{-1}) = 1 + d_1 q^{-1}$$
$$F(q^{-1}) = 1 + f_1 q^{-1}$$

and a zero-memory nonlinear power polynomial

$$w(t) = g_1 v(t) + g_3 v^3(t) + g_5 v^5(t).$$

The exerted systems input (field voltage e_f) has been a zero-mean, uniformly distributed, random sequence. The obtained response (armature terminal voltage e_a) has been corrupted by a zero-mean Gaussian noise sequence representing measurements errors. Based on Monte Carlo simulation of 10 runs, PE and GA estimates are presented, with NSR of 10%, in Table 9.8.

Algorithm	PE	GA
No. samples	500	100
No. iterations	50	-
No. generations	-	200
a_1	-0.9632 ± 0.0149	-0.8520 ± 0.0118
a_2	0.0433 ± 0.0153	-0.0613 ± 0.0244
b_1	0.5237 ± 0.2442	0.0467 ± 0.1511
c_1	-0.5933 ± 0.6987	-0.1629 ± 0.2444
c_2	-0.0795 ± 0.4515	0.3340 ± 0.0760
d_1	-0.1602 ± 0.4975	-0.0937 ± 0.1934
f_1	0.4516 ± 0.0130	0.5435 ± 0.0326
g_1	0.0348 ± 0.0096	0.0199 ± 0.0270
g_2	-0.0001 ± 0.0000	-0.1698 ± 0.2085
g_3	0.0000 ± 0.0000	1.2756 ± 0.1234

Table 9.8. Monte Carlo results for the nonlinear DC generator (10 experiments).

9.3.1 Model validation

Model validation is an important step in system identification since it is often the final check on the goodness of fit of any identified model. A number of methods have been developed for linear model validation [1]. Correlation based validation involves computing correlation functions composed of model residuals and system inputs and testing if these lie within preset confidence intervals.

Unfortunately, nonlinear model validation, including Wiener-Hammerstein models, has not received much attention as linear model validation. One of the very few efforts was proposed by Billings *et al* in [53] and [54]. They have developed nonlinear model validation based on NARMAX by extending the linear model based validation tests.

Using a 4-level pseudo-random input signal shown in Figure 9.19, the actual system output and the estimated Wiener-Hammerstein model output are generated and are shown as in Figures 9.20 and 9.21. It is evident from these figures that the Wiener-Hammerstein model outputs match very well the actual system output. The residuals, in terms of a logarithmic scale, are shown in Figure 9.22.

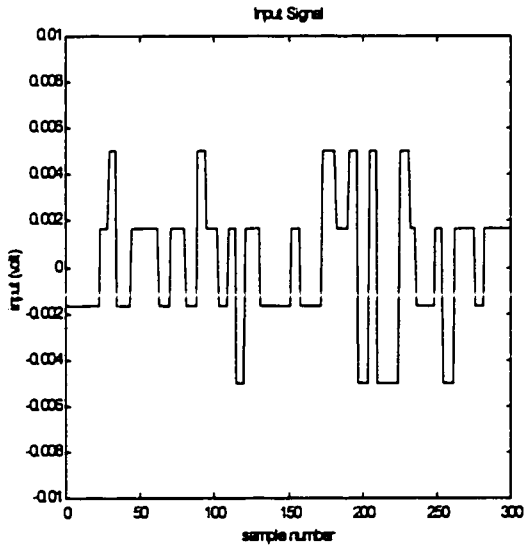


Figure 9.19. A 4-level pseudo-random input.

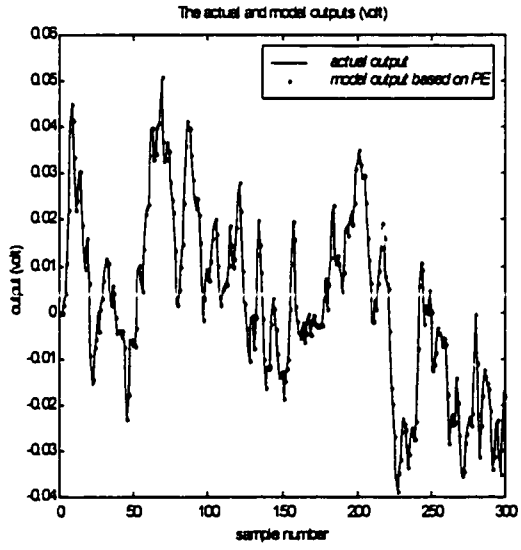


Figure 9.20. Actual and estimated PE outputs of the DC generator.

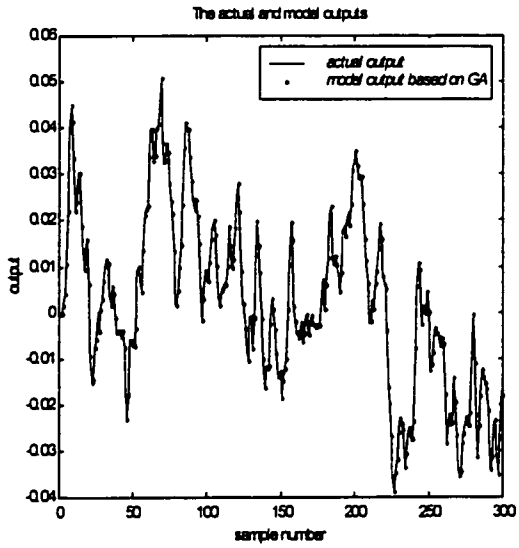


Figure 9.21. Actual and estimated GA outputs of the DC generator.

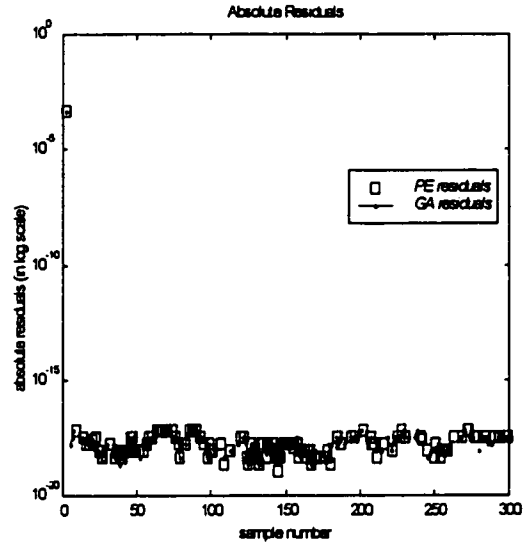


Figure 9.22. Absolute residuals in the outputs.

Furthermore, Figures 9.23 and 9.24 compare the actual step response and the step responses obtained based on PE and GA estimation. Figure 9.25 represents the residuals in the outputs in a logarithmic scale. Similarly, Figures 9.26, 9.27 and 9.28 show the frequency responses and their residuals.

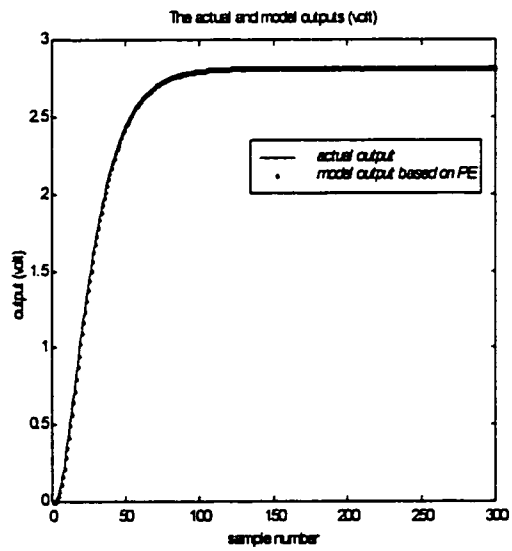


Figure 9.23. Actual and estimated PE step responses of the DC generator.

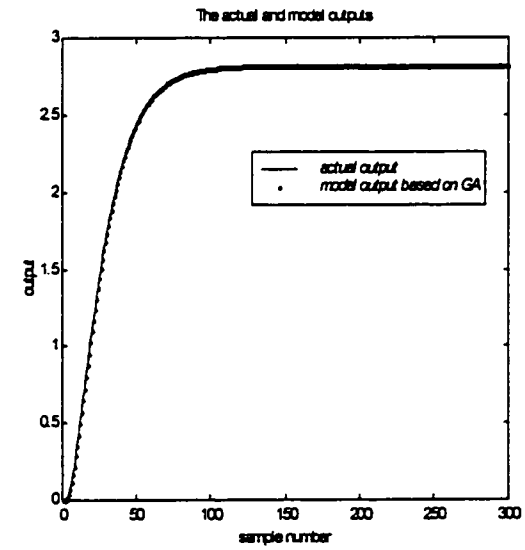


Figure 9.24. Actual and estimated GA step responses of the DC generator.

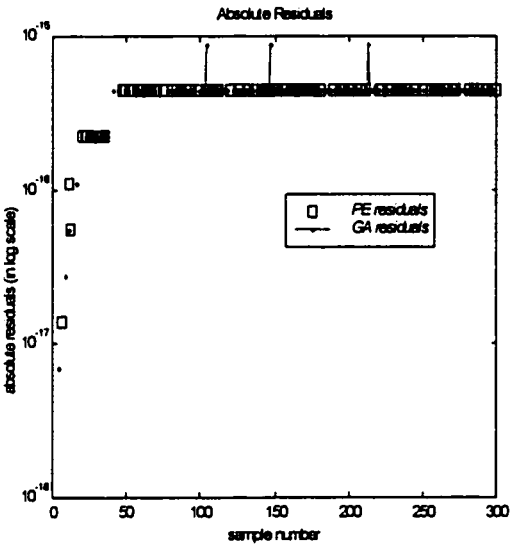


Figure 9.25. Absolute residuals in the step responses.

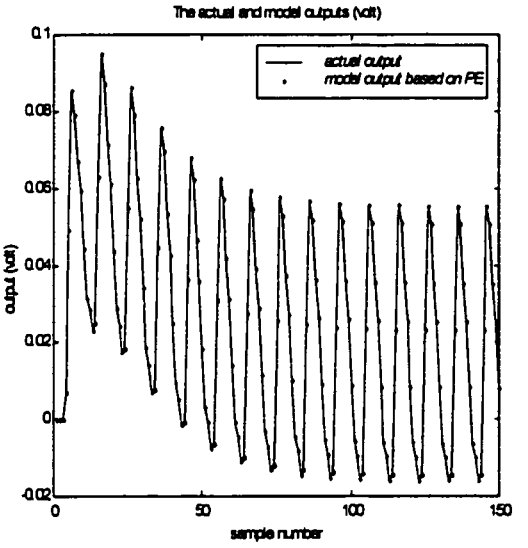


Figure 9.26. Actual and estimated PE frequency responses of the DC generator.

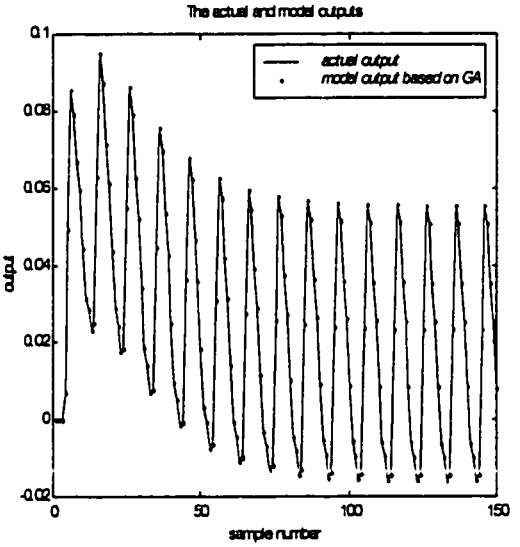


Figure 9.27. Actual and estimated GA frequency responses of the DC generator.

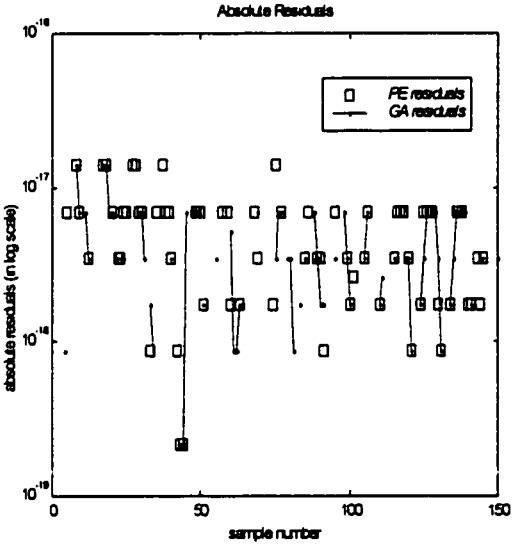


Figure 9.28. Absolute residuals in the frequency responses.

Figures 9.20 to 9.28 indicate the adequacy of the estimated models, based on PE or GA, in predicting the true system behavior. In the sequel, we will perform two validation tests. The first is investigating the autocorrelation of the prediction errors, and the second is the investigation of the crosscorrelation between the input and the prediction errors.

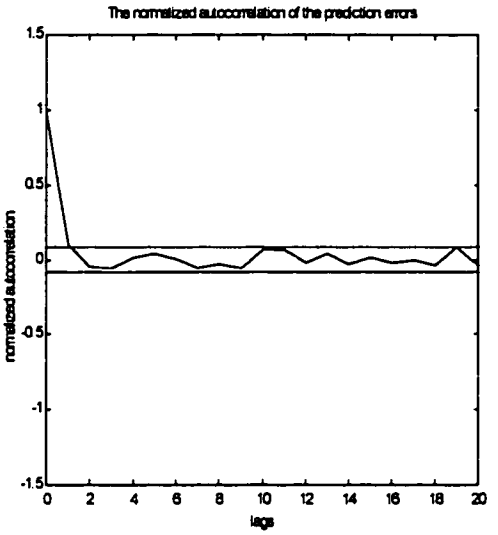


Figure 9.29. The normalized autocorrelation of the residuals.

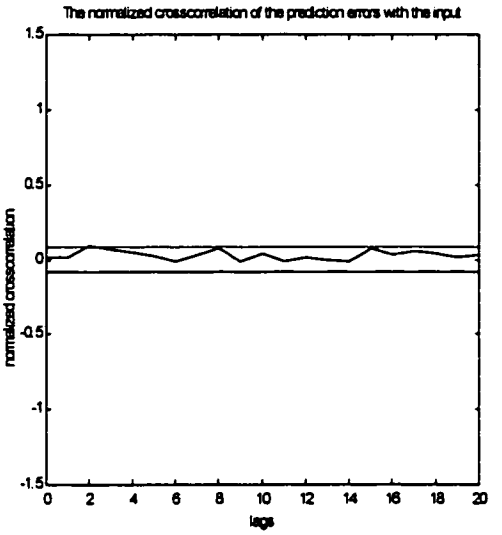


Figure 9.30. The normalized crosscorrelation of the input/residuals.

Figure 9.29 illustrates the normalized sample autocorrelation function of the prediction errors and a $\pm 1.96/\sqrt{N}$ band. The normalized sample crosscorrelation function between the input and the residuals and a 95% confidence interval are shown in Figure 9.30. Inspection of Figures 9.29 and 9.30 shows that the normalized correlation functions are well inside the 95% confidence bands indicating that the models obtained earlier can be considered as accurate models.

Chapter 10

CONCLUSIONS AND RECOMMENDATIONS

10.1 Conclusions

Nonlinear system identification via Wiener-Hammerstein model representations have been considered in this research. Unlike alternative approaches that are mainly in the correlation domains, the methods proposed herein are all time-domain based techniques. Both offline and online versions of Prediction Error, Output Error, Least Squares, and Instrumental Variable methods are developed to estimate the parameters of Wiener-Hammerstein models. Statistical properties of the estimates have been addressed both by analysis and variety of simulations.

The use of Genetic Algorithms in the above identification problem was also investigated. A Genetic Algorithm has been developed and utilized to estimates the Wiener-Hammerstein model parameters. In addition, the accuracy of such estimates are proved via simulation.

10.2 Contributions

The main contribution of the present work is the extension of the major techniques used for linear system identification to the class of nonlinear systems represented by Wiener-Hammerstein models. These extensions are thought to be suitable due to their wide use in linear system identification. However, due to presence of nonlinearity at both ends of the linear blocks, the extensions were not straight forward. Most of the proposed algorithms are different than the ones in the linear case, however, the basic principle is retained. In addition, an identification algorithm based on the Genetic Algorithms is also proposed. The proposed algorithms have the following features:

- The ability to estimate stochastic Wiener-Hammerstein model parameters that explicitly account for both the input/output and noise dynamics.
- The ability to perform identification from a single, even short, record of data corrupted by a realistic noise to signal ratio (NSR); a very significant advantage for many practical applications.
- The ability to operate directly on time-domain and simultaneously estimate all of the system parameters. Errors and huge computations related to the evaluation of

correlation functions, Fourier transforms, and multi-stage procedures are effectively avoided, while *a priori* system information may be directly incorporated into the identification procedure.

- The ability to accurately apply the new evolutionary techniques, presented by the GAs, in the Wiener-Hammerstein models identification problem.

10.3 Recommendations for Further Work

Obtaining an accurate model is a difficult job. Once an adequate model has been established, a control procedure can be applied with more ease. Therefore, one obvious extension of this work is to perform control laws applied to linear system to the Wiener-Hammerstein models identified by the algorithms presented herein. An important advantage of the Wiener-Hammerstein model and their subclasses, namely the Hammerstein models and the Wiener models, is that they permit the use of standard linear controller design methods. This is possible because the static nonlinearity in the process can be cancelled by inserting the nonlinear inverse of the static nonlinearity in the appropriate location in the loop. For the Hammerstein models, the inverse element should be located at the output of the controller. For the Wiener models, the inverse should be located at the output of the process. Thus the controller sees only the linear dynamic part of the process, and hence conventional linear controller synthesis methods can be used. In this regard, Roux *et al* [55] have

investigated an adaptive predictive control algorithm based on Hammerstein models and applied it in a continuous-flow fermentation process.

The inclusion of time-delays or time-varying parameters in the Wiener-Hammerstein model structure, the investigation of the proper selection of the initial values of the search algorithms given in this research, and the optimization of the GAs operators are all issues that can be studied by a future work. Another possible extension for further research will be studying the multivariable versions of the Wiener-Hammerstein models.

Appendix A

PSEUDOCODES FOR THE GENETIC ALGORITHM

A.1 Definitions and Notations

N = number of input/output data pairs.

p = population size.

P_old = a random initial population, whose values are taken from *U(0,1)*. It is scaled such that all the parameters fall in their spaces.

P_new = the population after applying the genetic operator, i.e. crossover.

nop = number of parameters per string.

rand= a random number generated from *U(0,1)*.

θ_i = a candidate parameter vector in the population.

i = string, or parameter vector, index in the population. $1 \le i \le p$

n = parameter index in the string. $1 \le n \le nop$

n_{low}, and n_{high} represent the search space of parameter n.

k = number of generations to run the algorithms over.

s = number of selected strings in the reproduction phase.

$\theta_i^{best}(k)$ = best string in generation k.

$J(\theta_i)$ = cost function. It is chosen as $J(\theta_i) = \sum_{j=1}^{p} (\varepsilon(j))^2$, where ε is the predicted error value using the model with string i.

average_J=mean(J).

$f(\theta_i)$ = fitness function. $f(\theta_i) = \dfrac{1}{J(\theta_i)+1}$ for each string i. $0 < f(\theta_i) < 1.0$.

$f_n(\theta_i)$ = normalized fitness value. $f_n(\theta_i) = \dfrac{f(\theta_i)}{\sum_{i}^{p} f(\theta_i)}$ of each string i.

$F(\theta_i)$ = cumulative normalized fitness. $F(\theta_i) = \sum_{x=1}^{i} f_n(x)$ for a population, index used is string i.

P_p =The perturbation probability. P_c =The crossover probability. P_m =The mutation probability.

A.2 General GA Algorithm

procedure *GA*
begin
    initialize population *P(0)*
    evaluate *P(0)*
    k=1
    repeat
        select *P(k)* from *P(k-1)*
        recombine *P(k)*
        evaluate *P(k)*
        fill *P(k)*
    until (termination condition)
end

A.3 Scaling

for *i=1* **to** *p* **do**
    for *n=1* **to** *nop* **do**
        $P\_new(n,\theta_i) = \left(n_{high} - n_{low} \right) \times P\_old(n,\theta_i) + n_{low}.$
    end do
end do

A.4 Evaluation

for *i*=1 **to** *p* **do**

$$J(\theta_i) = \sum_{j=1}^{p} (\varepsilon(j))^2$$

$$f(\theta_i) = \frac{1}{J(\theta_i)+1}$$

$$f_n(\theta_i) = \frac{f(\theta_i)}{\sum_{i}^{p} f(\theta_i)}$$

$$F(\theta_i) = \sum_{x=1}^{i} f_n(x)$$

end do

$$average\_J = \frac{1}{p}\sum_{i=1}^{p} j(\theta_i)$$

A.5 Monotonicity Test

if $f(\theta_i^{best}(k-1)) \le f(\theta_i^{best}(k))$ **then**

$\theta_i^{best}(k) = \theta_i^{best}(k-1)$

end if

A.6 Perturbed Reproduction Based on RWS (Roulette Wheel Selection)

for m=1 **to** s **do**

if $F(\theta_{i-1}) \le rand \le F(\theta_i)$ **then**

$P\_new(:,m)=P\_old(:,\theta_i)$

if $rand \le P_p$ **then**

$\sigma^2 = 1 - f(\theta_i)$

$P\_new(:,m)=P\_new(:,\theta_i) + N(0,\sigma^2)$

end if

end if

end do

A.7 Crossover

for m=1 **to** s **do**
  if $rand \le P_c$ **then**

$\theta_1 = fix((s-1) \times rand + 1)$

$\theta_2 = fix((s-1) \times rand + 1)$

$r = fix((nop-1) \times rand + 1)$

$$P\_new(:,\theta_1) = \begin{pmatrix} I_{r\times r} & 0 \\ 0 & 0 \end{pmatrix} \times P\_old(:,\theta_1) + \begin{pmatrix} 0 & 0 \\ 0 & I_{(nop-r)\times(nop-r)} \end{pmatrix} \times P\_old(:,\theta_2)$$

$$P\_new(:,\theta_2) = \begin{pmatrix} 0 & 0 \\ 0 & I_{(nop-r)\times(nop-r)} \end{pmatrix} \times P\_old(:,\theta_1) + \begin{pmatrix} I_{r\times r} & 0 \\ 0 & 0 \end{pmatrix} \times P\_old(:,\theta_2)$$

  end if
end do

A.8 Mutation Definition

Step 1. $P\_old(n,\theta_i) = I_1 I_2 . F_1 F_2 F_3 F_4$

Step 2. $temp1 = P\_old(n,\theta_i) - I_1 I_2 . 0 F_2 F_3 F_4 = 00 . F_1 000$

Step 3. $temp2 = P\_old(n,\theta_i) - I_1 I_2 . F_1 0 F_3 F_4 = 00.0 F_2 00$

Step 3. $\mathrm{M}(P\_old(n,\theta_i)) = 0.1 \times temp1 + 10 \times temp2 + I_1 I_2 . 00 F_3 F_4$

A.9 Mutation

for m=1 **to** s **do**
  if $rand \le P_m$ **then**

$r_2 = fix((nop-1) \times rand + 1)$

$\theta_4 = fix((s-1) \times rand + 1)$

if $J(i) \le average\_J$

$P\_new(r_2,\theta_4) = \mathbf{M}(P\_old(r_2,\theta_4))$

else

$P\_new(r_2,\theta_4) = -P\_old(r_2,\theta_4)$

```
            end if
    end if
end do
```

A.10 Filling

for $m=s+1$ **to** p **do**

$$P\_new = [P\_new \quad rand \times \left(\theta_i^{high} - \theta_i^{low} \right) + \theta_i^{low}]$$

end do

Appendix B

MATRIX INVERSION LEMMA

Provided the inverses below exist,

$$[A+BCD]^{-1} = A^{-1} - A^{-1}B[C^{-1} + DA^{-1}B]^{-1}DA^{-1} \tag{B.1}$$

Proof. The statement is verified by direct multiplication.

$$\begin{aligned}
&[A+BCD]\{A^{-1} - A^{-1}B[C^{-1} + DA^{-1}B]^{-1}DA^{-1}\} \\
&\quad = I + BCDA^{-1} - [B + BCDA^{-1}B][C^{-1} + DA^{-1}B]^{-1}DA^{-1} \\
&\quad = I + BCDA^{-1} - BC[C^{-1} + DA^{-1}B][C^{-1} + DA^{-1}B]^{-1}DA^{-1} \\
&\quad = I
\end{aligned}$$

which proves (B.1)

Appendix C

ESSENTIAL THEOREMS AND DEFINITIONS

Theorem C.1

Consider

$$X_N = \frac{1}{\sqrt{N}} \sum_{t=1}^{N} z(t) \tag{C.1}$$

where $z(t)$ is a (vector-valued) zero mean stationary process given by

$$z(t) = \phi(t)\upsilon(t) \tag{C.2}$$

In (C.2), $\phi(t)$ is a matrix and $\upsilon(t)$ is a vector. The entries of $\phi(t)$ and $\upsilon(t)$ are stationary, possibly correlated, ARMA processes with zero means and underlying white noise sequences with finite fourth-order moments. The elements of $\phi(t)$ may also contain a bounded deterministic term.

Assume the limit

$P = \lim_{N\to\infty} E\{X_N X_N^{\mathrm{T}}\}$ exists and is nonsingular. Then X_N is asymptotically Gaussian distributed,

$$X_N \xrightarrow{dist} \mathrm{N}(0, P) \tag{C.3}$$

Proof. See [1]

Definition C.1

Let A be an $(m \mid n)$ matrix, and let a_i denote its ith column.

$A = (a_1 \cdots a_n)$

Then the $(mn \mid 1)$ column vector $\mathrm{vec}(A)$ is defined as

$$\mathrm{vec}(A) = \begin{pmatrix} a_1 \\ a_2 \\ \vdots \\ a_n \end{pmatrix}$$

Appendix D

GLOSSARY

This Appendix lists some basic definitions and properties used in this thesis.[1,2,3]

D.1 Random Process

A random (stochastic) process is a collection of all time-history records (sample records or ensemble) that the random phenomenon might have produced. Random processes may either be stationary or nonstationary.

D.2 Sample Record

A sample record is a single time history record, observed over a finite time interval, which is a possible realization of the random phenomenon.

D.3 Stationary Random Process

A stationary random process is a process that has statistical properties which are invariant with respect to translations in time. To explain this, consider for example the random process $\{x(t)\}$. The mean value (first moment) $\mu_x(t_1)$ and the autocorrelation function (second moment) $R_{xx}(t_1,t_1+\tau)$ of the random process at some time t_1 are given by

$$\mu_x(t_1) = \lim_{N\to\infty} \frac{1}{N}\sum_{k=1}^{N} x_k(t_1) \tag{D.1}$$

$$R_{xx}(t_1,t_1+\tau) = \lim_{N\to\infty} \frac{1}{N}\sum_{k=1}^{N} x_k(t_1)x_k(t_1+\tau) \tag{D.2}$$

when $\mu_x(t_1)$ and $R_{xx}(t_1,t_1+\tau)$ do not vary as time t_1 varies, then the process is said to be stationary. That is, for a stationary random process, the mean value is a constant $(\mu_x(t_1) = \mu_x)$ and the autocorrelation function is dependent only on the time shift τ, i.e. $(R_{xx}(t_1,t_1+\tau) = R_{xx}(\tau))$.

D.4 Gaussian Random Variable

A random variable $x(t)$ is said to be Gaussian (or normal) when its probability density function at any time t is given by

$$p(x) = \frac{1}{\sigma_x \sqrt{2\pi}} e^{\left[\frac{-(x-\mu_x)^2}{2\sigma_x^2}\right]} \tag{D.3}$$

where μ_x and σ_x are, respectively, the mean value and the standard deviation of the random variable $x(t)$ given by

$$\mu_x = E\{x(t)\} = \int_{-\infty}^{\infty} xp(x)dx \tag{D.4}$$

$$\sigma_x^2 = E\{x^2(t)\} - \mu_x^2 \tag{D.5}$$

where $E\{.\}$ stands for the expected value.

D.5 Autocorrelation Function

The autocorrelation function $R_{xx}(\tau)$ of a random process $x(t)$ is the average of the product of $x(t)$ and $x(t+\tau)$ for reasonably large averaging time T. That is

$$R_{xx}(\tau) = E\{x(t)x(t+\tau)\} = \lim_{T\to\infty} \frac{1}{T} \int_0^T x(t)x(t+\tau)dt \tag{D.6}$$

For a discrete sample record $x(k)$, the autocorrelation function $R_{xx}(\tau)$ is give by

$$R_{xx}(\tau) = E\{x(k)x(k+\tau)\} = \lim_{T\to\infty} \frac{1}{N-\tau} \sum_{k=1}^{N-\tau} x(k)x(k+\tau) \tag{D.7}$$

where N is the length of the record.

D.6 Crosscorrelation Function

The crosscorrelation function $R_{xy}(\tau)$ of two random processes $x(t)$ and $y(t)$ is the average of the product of $x(t)$ and $y(t+\tau)$ for reasonably large averaging time T. That is

$$R_{xy}(\tau) = E\{x(t)y(t+\tau)\} = \lim_{T\to\infty} \frac{1}{T} \int_0^T x(t)y(t+\tau)dt \tag{D.8}$$

In discrete form, $R_{xy}(\tau)$ is give by

$$R_{xy}(\tau) = E\{x(k)y(k+\tau)\} = \lim_{T\to\infty} \frac{1}{N-\tau} \sum_{k=1}^{N-\tau} x(k)y(k+\tau). \tag{D.9}$$

D.7 Monicity of Polynomial

Let $D(s)$ be a polynomial with real coefficients expressed as

$$D(s) = D_n s^n + D_{n-1} s^{n-1} + \cdots + D_1 s + D_0 \tag{D.10}$$

The polynomial $D(s)$ is said to be of *degree* n if $D_n \neq 0$. The coefficient D_n associated with the highest power of s is called the *leading coefficient*. A polynomial is called *monic* if its leading coefficient is equal to 1.

D.8 Multiple Taylor Series Expansion

a. Given a real function $f(x_1, x_2, \cdots, x_n)$ such that all partial derivatives of order m exist and are continuous for $a_i \le x_i \le b_i$ $(i = 1, 2, \cdots, n)$, one has

$$\begin{aligned} f(x_1, x_2, \cdots, x_n) = f(a_1, a_2, \cdots, a_n) &+ \sum_{i=1}^{n} \left.\frac{\partial f}{\partial x_i}\right]_{a_1, a_2, \cdots, a_n} (x_i - a_i) \\ &+ \frac{1}{2!} \sum_{i=1}^{n} \sum_{j=1}^{n} \left.\frac{\partial^2 f}{\partial x_i \partial x_j}\right]_{a_1, a_2, \cdots, a_n} (x_i - a_i)(x_j - a_j) + \cdots \\ &+ R_m(x_1, x_2, \cdots, x_n) \qquad (a_i \le x_i < b_i; i = 1, 2, \cdots, n) \end{aligned} \tag{D.11}$$

In terms of differentials

$$\begin{aligned} f(a_1 + dx_1, a_2 + dx_2, \cdots, a_n + dx_n) &- f(a_1, a_2, \cdots, a_n) \\ &= df + \frac{1}{2!} d^2 f + \cdots + \frac{1}{(m-1)!} d^{m-1} f + R_m \end{aligned} \tag{D.12}$$

where all differentials are computed at the point $(a_1, a_2, \cdots, a_n)$, and

$$R_m = \left.\frac{1}{m!} d^m f\right]_{X_1, X_2, \cdots, X_n} \qquad (a_i < X_i < a_i + dx_i; i = 1, 2, \cdots, n) \tag{D.13}$$

b. If all derivatives of $f(x_1, x_2, \cdots, x_n)$ exist and $\lim_{m \to \infty} R_m(x_1, x_2, \cdots, x_n) = \mathbf{0}$ for $a_i \le x_i < b_i$, then equation (D.11) yields a multiple power-series expansion for $f(x_1, x_2, \cdots, x_n)$ (*multiple Taylor series*). [51]

D.9 Bias, Convergence and Consistency

Bias. An estimate $\hat{\theta}$ is *biased* if its expected value deviates from the true value, i.e.

$$E\{\hat{\theta}\} \neq \theta_0 \tag{D.14}$$

The difference $E\{\hat{\theta} - \theta_0\}$ is called *the bias.* If instead equality applies in equation (D.14), $\hat{\theta}$ is said to be *unbiased.*

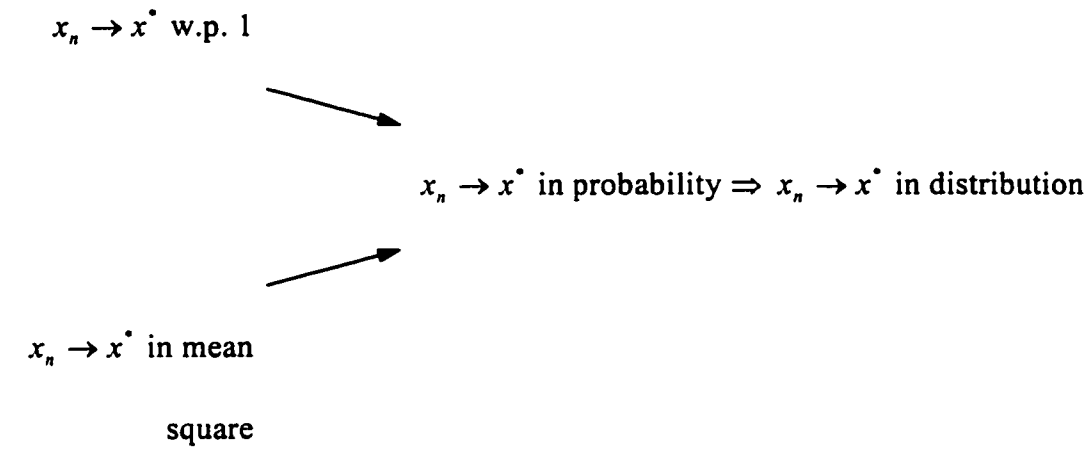
Convergence. Let $\{x_n\}$ be an indexed sequence of stochastic variables. Let x^* be a stochastic variable. Then

- $x_n \to x^*$ (as $n \to \infty$) *with probability one* (w.p. 1) if $P(x_n \to x^*, n \to \infty) = 1$.
- $x_n \to x^*$ *in mean square* if $E\{x_n - x^*\}^2 \to 0, n \to \infty$.
- $x_n \to x^*$ *in probability* if for every $\varepsilon > 0, P\left(|x_n - x^*| > \varepsilon\right) \to 0$ as $n \to \infty$.
- $x_n \to x^*$ *in distribution* if $f_{x_n}(x) \to f_{x^*}(x)$ where $f_{x_n}(x)$, $f_{x^*}(x)$ denote the probability density functions of x_n and x^*, respectively.

Remark. If $f_{x^*}(x)$ is the Gaussian distribution $N(m,P)$ and $x_n \to x^*$ in distribution it is said that x_n is asymptotically Gaussian distributed, denoted by

$$x_n \overset{dist}{\to} N(m,P)$$

The following connections exist between these convergence concepts:



Consistency. We say that an estimate $\hat{\theta}$ is *consistent* if

$$\hat{\theta} \to \theta_0 \text{ as } N \to \infty \tag{D.15}$$

Since $\hat{\theta}$ is a stochastic variable, it must be defined in what sense the limit in (D.15) should be taken. A reasonable choice is 'limit with probability one'.

D.10 Euclidean Norms

The Euclidean vector norm for $\mathbf{x} = (x_1, x_2, \cdots, x_n)$ is defined as

$$\|\mathbf{x}\| = (x_1^2 + \cdots + x_n^2)^{1/2} \tag{D.16}$$

The Euclidean norm of a matrix norm is defined as follows. Let A be a 'rectangular' $m \times n$ matrix, i.e.

$$A \equiv \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & \cdots & \cdots & a_{1n} \\ \cdots & \cdots & \cdots & \cdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{pmatrix} \equiv (a_{ik})$$

Then, the Euclidean norm of A is

$$\|A\|_2 = \left(\sum_{i=1}^{m} \sum_{k=1}^{n} |a_{ik}|^2 \right)^{1/2}$$

References

1. Soderstrom, T. and P. Stoica, (1989). *System Identification*. Prentice-Hall.
2. Goodwin, G. and K. Sin, (1984). *Adaptive Filtering, Prediction and Control*. Prentice-Hall, Englewood Cliffs.
3. J. Norton, (1986). *An Introduction to Identification*. Academic Press.
4. Haber, R. and Unbehauen. "Structure identification of nonlinear dynamic systems-A survey on input/output approaches." *Automatica*, 1990, vol. 26, no. 4, pp. 651-677.
5. Poncet, A. and G. Moschytz. "Selecting inputs and measuring nonlinearity in system identification." *Proceedings of International Workshop on Neural Networks for Identification, Control, Robotics and Signal/Image Processing, Venice, Italy*, 21-23 Aug. 1996, pp-2-10.
6. Emara-Shabaik, H., K. Moustafa, and J. Talaq. "Structure identification of a class of nonlinear systems using correlation and bispectrum approaches." *UKACC International Conference on Control '96, 2-5 Sep. 1996,* pp.246-251.
7. Emara-Shabaik, H., J. Bomberger, and D. Seborg. "Cumulant / bispectrum model structure identification applied to a pH neutralization process." *UKACC International Conference on Control '96, 2-5 Sep. 1996,* pp.1046-1051.
8. Ljung, L. and T. Soderstrom, (1983). *Theory and Practice of Recursive Identification*. Cambridge, MA: MIT Press.
9. M. Schetzen, (1980). *The Volterra and Wiener Theories of Nonlinear Systems*. John Wiley & Sons, Inc.
10. Billings, S. and W. Voon. "A Predictive-error and stepwise-regression estimation algorithm for nonlinear systems." *Int. J. Control*, 1986, vol. 44, no. 3, pp. 803-822.

11. Soufian, M., M. S. and M. Thomson. "Practical comparison of neural networks and conventional identification methodologies." *Artificial Neural Networks Conference*, 7-9 July 1997, pp. 262-267.
12. Vandersteen, G. and J. Schoukens. "Measurements and identification of nonlinear systems consisting out of linear dynamic blocks and one static nonlinearity." *IEEE Instrumentation and Measurement Technology Conference. Ottawa, Canada*, May 19-21, 1997, pp. 853-858.
13. Hsu, J.-T. and Khai D. "Behavioral modeling of the IGBT using the Hammerstein configuration." *IEEE Trans. Power Electronics*, 1996, vol. 11, no. 6, 746-754.
14. Kearney, R., R. Stein, and L. Parameswaran. "Identification of intrinsic and reflux contributions to human ankle stiffness dynamics." *IEEE Trans. Biomedical Engineering*, 1997, vol. 44, no. 6, 493-504.
15. P. Young. "Parameter estimation for continuous-time models-a survey." *Automatica*, 1981, vol. 17, no. 1, pp. 23-39.
16. Yao, L. and W. Sethares. "Nonlinear parameter estimation via Genetic Algorithms." *IEEE Transactions on Signal Processing*, April 1994, vol. 42, no. 4, pp. 927-935.
17. Kristinsson, K. and G. Dumont. "System identification and control using genetic algorithms." *IEEE Transactions on Systems, Man, and Cybernetics*, Sep./Oct. 1992, vol. 22, no. 5, pp. 1033-1046.
18. Narendra, K. S. and P. G. Gallman. "An iterative method for the identification of nonlinear systems using a Hammerstein model." *IEEE Trans. Automatic Control*, AC-11, 1966, no. 3, 546-560.
19. Chang, F. H. and R. Luus. "A noniterative method for identification using the Hammerstein model." *IEEE Trans. Automatic Control*, AC-16, 1971, no. 5, pp. 464-468.
20. Kortmann, M. and H. Unbehauen. "Identification methods for nonlinear MISO systems." *IFAC Triennial World Congress, Munich*, 1987, pp. 233-238.
21. Haber, R. "Parameter identification of nonlinear dynamic systems based on correlation functions." *5th IFAC Symposium on Ident. And Sys. Param. Estim., Darmstadt*, 1979, 515-522.

22. P. Stoica. "On the convergence of an iterative algorithm used for Hammerstein system identification." *IEEE Trans. Automatic Control.* 1981, vol. 26, no. 4, pp. 967-969.

23. Rangan, S., G. Wolodkin and K. Poolla. "New results for Hammerstein system identification." *Proc. 34th Conf. Decision and Control*, Dec. 1995, pp. 697-702.

24. Ralston, J. and M. Zoubir. "Identification of time-varying Hammerstein systems with application to knock detection in combustion engines." *IEEE Annual Conference on Speech and Image Technologies for Computing and Telecommunications*, 1997, vol. 2, pp. 799-802.

25. Hu, D. and Z. Wang. "An identification method for the Wiener model of nonlinear systems." *Proc. 30th Conf. Decision and Control*, Dec. 1991, pp. 783-787.

26. T. Wigren. "Recursive prediction error identification using the nonlinear Wiener model." *Automatica*, 1993, vol. 29, no. 4, pp. 1011-1025.

27. T. Wigren. "Convergence analysis of recursive identification algorithms based on the nonlinear Wiener model." *IEEE Trans. Automatic Control.* 1994, vol. 39, no. 11, pp. 2191-2206.

28. Kalafatis, A., L. Wang and W. Cluett. "Identification of Wiener-type nonlinear systems in a noisy environment." *Int. J. Control,* 1997, vol. 66, no. 6, pp. 923-941.

29. Billings, S. A. and S. Y. Fakhouri. "Identification of a class of nonlinear systems using correlation analysis." *Proc. Instn. Elect. Engrs*, 1978, vol. 125, no. 7, pp. 691-697.

30. Billings, S. A. and S. Y. Fakhouri. "Nonlinear systems identification using the Hammerstein model." *Int. J. Systems Science*, 1979, vol. 10, no. 5, pp. 567-578.

31. Yoshine, K. and N. Ishii. "Nonlinear analysis of a linear-non-linear-linear system." *Int. J. Systems Sci*, 1992, vol. 23, no. 4, pp. 623-630.

32. Boutayeb, M. and M. Darouach. "Recursive identification method for MISO Wiener-Hammerstein model." *IEEE Trans. Automatic Control*, 1995, vol. 40, no. 2, pp. 287-291.

33. Emara-Shabaik and K. Moustafa. "Structure identification criteria of nonlinear systems via bispectrum." *The Arabian Journal for Science and Engineering*, July 1995, vol. 20, no. 3, pp. 535-546.

34. J. H. Holland, (1975). *Adaptation in natural and artificial systems*. University of Michigan Press, Ann Arbor.

35. Caponetto, R., L. Fortuna, S. Graziani, and M. Xibilia. "genetic algorithms and applications in systems engineering: a survey." *Trans. Inst Meas. Control*, 1993, vol. 15, no. 3, pp. 143-156.

36. D. E. Goldberg, (1989). *Genetic algorithms in search, optimization and machine learning*. Addison-Wesly.

37. Kristinsson, K. and G. Dumont. "Genetic algorithms in system identification." *Third IEEE Int. Symp. Intelligent Control, Arlington, VA*, 1988, pp. 597-602.

38. Sheta, A. and K. DeJong. "Parameter estimation of nonlinear systems in noisy environments using genetic algorithms." *IEEE Int. Symp. Int. Control*, Dearborn, MI, Sep. 1996, pp. 360-365.

39. Billings, S. and K. Mao. "Structure detection for nonlinear rational models using genetic algorithms." *International Journal of Systems Science*, 1998, vol. 29, no. 3, pp. 223-231.

40. Etter, D., M. Hicks, and K. Cho. "Recursive adaptive filter design using an adaptive genetic algorithm." *IEEE International conference on Acoustics, Speech and Signal Processing*, 1982, vol. 2, pp. 635-638.

41. J. Grefenstette. "Optimization of control parameters for genetic algorithms." *IEEE Transactions of Systems, Man, and Cybernetics*, 1986, vol. 16, no. 1, pp. 122-128.

42. Lansberry, J., L. Wozniak, and D. Goldberg. "Optimal hydrogenerator governor tuning with a genetic algorithm." *IEEE Trans. on Energy Conversion*, 1992, vol. 7, no. 4, pp. 623-630.

43. P. Eykhoff, (1974). *System Identification*. John Wiley & Sons, Inc.

44. K. Godfrey. "Introduction to binary and nonbinary signals used in system identification." *IEE Colloquium on Multifrequency Testing for System Identification*, 1990, pp. 1-2.

45. H. Barker. "Design of multilevel pseudorandom signals for specified harmonic content." *International Conference on Control*, 1991, vol. 1, pp. 562-566.

46. M. Darnell. "Multi-level signals with good autocorrelation properties." *International Conference on Control*, 1991, vol. 1, pp. 556-561.

47. Nowak, R. and B. Van Veen. "Nonlinear system identification with pseudorandom multilevel excitation sequences." *IEEE International conference on Acoustics, Speech and Signal Processing*, 1993, vol. 4, pp. 456-459.

48. Glad, S. and L. Ljung. "Model structure identifiability and persistence of excitation." *Proc. 29th Conf. Decision and Control*, Dec. 1990, pp. 3236-3240.

49. Astrom, K. and B. Wittenmark, (1989). *Adaptive Control*. Addison-Wesley Publishing Company.

50. Arabas, J., Z. Michalewicz, and J. Mulawka. "GAVaPS - a genetic algorithm with varying population size." *IEEE First Conf. Of Evolutionary Computation*, 1994, vol. 1, pp. 73-78.

51. Korn, G. and T. Korn, (1968). *Mathematical Handbook for Scientists and Engineers*. McGraw-Hill Inc.

52. Chen, C.-H. and S. Fassois. "On the estimation of stochastic Wiener-Hammerstein-type systems with non-smooth non-linearity." *American Control Conference*, June 1997, pp. 1494-1498.

53. Billings, S. and Q. Zhu. "Nonlinear model validation using correlation tests." *Int. J. Control*, 1994, vol. 60, no. 6, pp. 1107-1120.

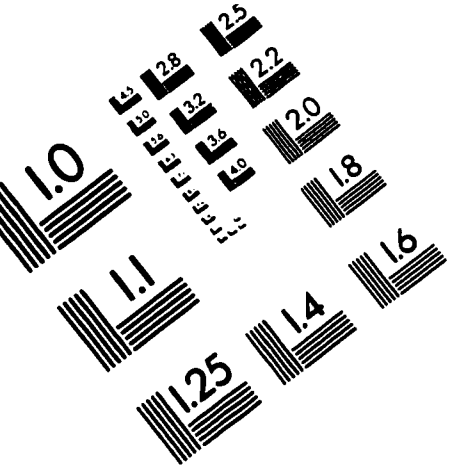
54. Billings, S. and W. Voon. " Correlation based model validity tests for non-linear models." *Int. J. Control*, 1986, vol. 44, no. 1, pp. 235-244.

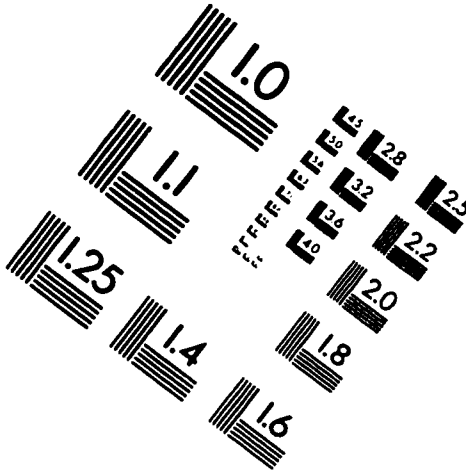
55. Roux, G., B. Dahhou, and Y. Sevely. " On the influence on GPC performances of nonlinear fermentation processes." *Int. Conf. Sys., Man and Cybernetics*, 1993, vol. 5, pp. 260-265.

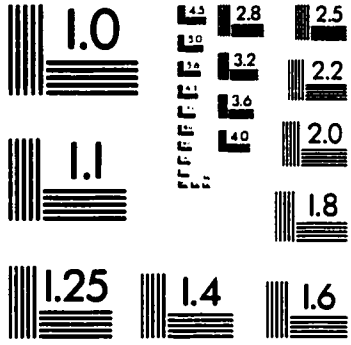
Vita

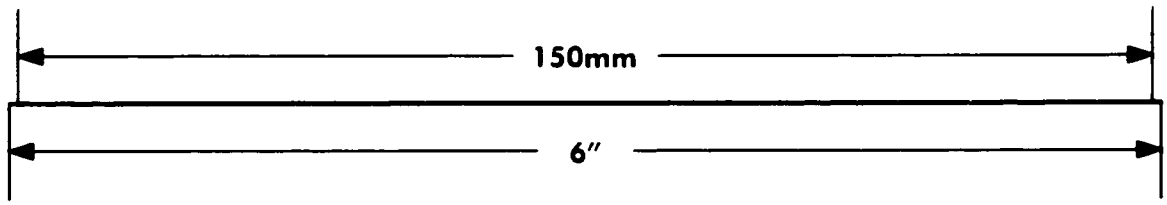
- Khaled Husain Al-Ajmi
- Born in Abha, Saudi Arabia on March 19, 1974.
- Received Bachelor's degree in Systems Engineering from King Fahd University of Petroleum and Minerals (KFUPM), Dhahran, Saudi Arabia in June 1996.
- Completed Master's degree requirements at KFUPM in April 1999.

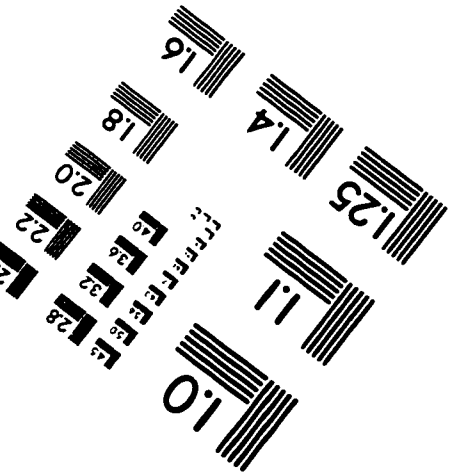
IMAGE EVALUATION
TEST TARGET (QA-3)

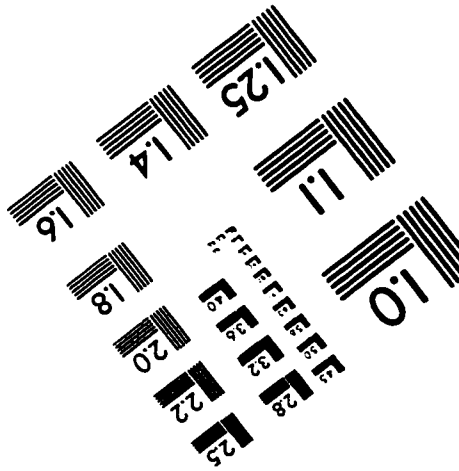












APPLIED IMAGE, Inc
1653 East Main Street
Rochester, NY 14609 USA
Phone: 716/482-0300
Fax: 716/288-5989